

Database Application Development • More GUI Controls • Intro to DSOM • Extending REXX

January/February 1994

Display Until 2/28/94

os/2 Developer

THE MAGAZINE FOR ADVANCED SOFTWARE DEVELOPMENT

Developing Database Applications

Database Tools

DB2/2—Is It for You?

Visual REXX
Database Prototyping

Whatever Happened
to Database Manager?

os/2
Lights Your Way
to Successful
Database
Development

U.S. \$9.95 Vol. 6 No. 1



02

WATCOM™

Visual Development Environment For

WATCOM VX•REXX is an easy to use visual development environment for creating applications that leverage the capabilities of OS/2 2.x and exploit the Presentation Manager graphical user interface. VX•REXX combines a project management facility, visual designer and an interactive source-level debugger to deliver a very approachable and highly productive visual development environment.

Design Applications Visually

Create rich graphical applications quickly and easily using the visual design environment. With the visual designer, you can graphically create Presentation Manager interface objects, quickly customize their properties, and easily attach REXX procedures using powerful drag-and-drop programming techniques.

Integrated Development Environment

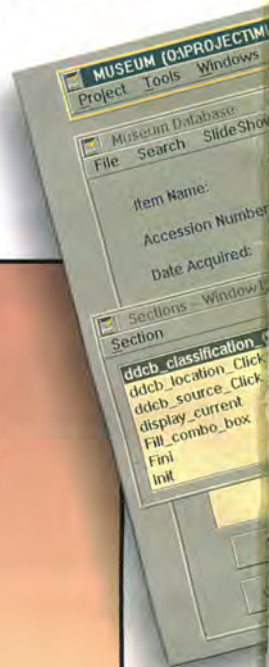
Build, test and debug your application without leaving the development environment. Then package your application as an EXE file or PM macro for royalty-free redistribution. The power of the integrated development environment and debugger can also be used with your existing REXX applications.

Powerful Open Environment

Enjoy the simplicity of event-driven programming together with the global editing capabilities essential for professional project management. WATCOM VX•REXX is open and extensible through IBM's object oriented System Object Model (SOM) technology. You can access all standard REXX API's including DB2/2, because VX•REXX is based on the OS/2 2.x standard system REXX.

Highlights

- ▶ Easy to use visual development environment
- ▶ Create and modify objects dynamically at both edit and run time
- ▶ Powerful project management facility
- ▶ Advanced interactive source-level debugger
- ▶ Package your applications as EXE files or PM macros
- ▶ Access to standard REXX API's including DB2/2
- ▶ System Object Model (SOM) based object manager
- ▶ Drag-and-drop programming
- ▶ Support for multi-threaded applications
- ▶ Include OS/2 style help and hints in your applications
- ▶ Supports SAA CUA'91 objects
- ▶ Integrated console window support for existing REXX programs
- ▶ Royalty-free run-time
- ▶ Multiple modeless window support
- ▶ Create PM macros for applications supporting REXX as a macro language



VX•REXX™

OS/2 REXX

Interactive Debugging

If an error occurs at run-time, VX•REXX will display a traceback pinpointing the source line where the error occurred. A simple click of the mouse will return you to the source edit window to correct the error. The built-in interactive source-level debugger lets you set breakpoints, step through code and watch variables to track down complex problems.

Build Professional Applications

WATCOM VX•REXX allows you to leverage key OS/2 features to create professional applications. Build applications that dynamically create and modify CUA'91 screen objects at both edit and run-time, and include OS/2 style help and hints.

Create Multi-Threaded Applications

Every VX•REXX application contains multiple threads. One thread remains responsive to user input while others continue processing. In addition, VX•REXX provides the ability for advanced applications to easily use additional threads.

Suggested Retail: \$199*

**Call Toll Free
1-800-265-4555**

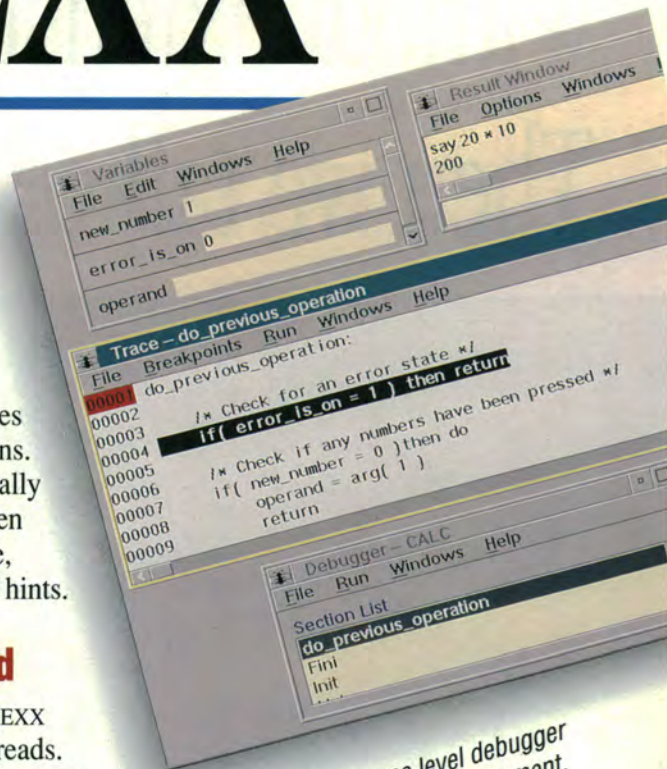
WATCOM

WATCOM International

415 Phillip Street, Waterloo, Ontario, Canada, N2L 3X2
Phone: (519) 886-3700 Fax: (519) 747-4971

*Prices and specification are subject to change without notice. Price does not include freight and taxes where applicable. Prices quoted in US dollars. WATCOM, the Lightning Device, and VX•REXX are trademarks of WATCOM International Corporation. Other trademarks are the properties of their respective owners. © Copyright 1993 WATCOM International Corporation.

Circle Reader Service Number 1



The integrated source level debugger
simplifies your project development.



WATCOM VX•REXX:

The integrated visual
solution builder for OS/2 2.x.

Project management facilities allow quick and easy
development of your OS/2 applications.

“We were told it was impossible to develop a client/server application without extensive retraining. Then we talked to Micro Focus.”



Larry Lowder, Systems Architect, Questar Service Corporation.

Mountain Fuel Supply®, a division of Questar®, is a utility company supplying natural gas to 750,000 customers across Utah, Wyoming, Idaho and Colorado. The company's success is largely driven by its implicit belief that the customer is number one.

Yet, IT also plays its part in that success: client/server architectures and graphical user interfaces (GUIs) have helped Mountain Fuel Supply move applications and information closer to the customers and the employees. All of which has resulted in an augmented level of service being offered to customers.

When Larry Lowder, one of Questar's Systems Architects, set out to build the client/server architecture for Mountain Fuel Supply, he needed solutions, not skepticism. For the first project, a cashiering system, he needed to link workstations with OS/2® to the DB2® database on the host, running CICS™

"We were faced with having to spend up to two years retraining our

COBOL programmers in C and API calls. Then we discovered Micro Focus Dialog System™. It allowed us to build the client functionality we required, and re-engineer the existing mainframe application as a server."

"Within a week, mainframe programmers were producing GUI screens for COBOL. Within 90 days we had delivered the system. Now we're not only coming in under budget, but also way ahead of schedule."

As Mountain Fuel Supply discovered, the Micro Focus solution lets you make a productive transition to client/server without sacrificing any of the resources you've worked so hard to build.

When the world's leading corporations demand "A Better Way of Programming," they turn to Micro Focus. For a brochure on putting the Micro Focus Client/Server Solution to work for you, call 800-872-6265.

MICRO FOCUS®

Micro Focus Inc. 2465 East Bayshore Road, Palo Alto, CA 94303. Tel. (415) 856 4161.

Micro Focus is a registered trademark and Dialog System and "A Better Way of Programming" are trademarks of Micro Focus, Inc. All other trademarks are property of their respective companies.

GSA Contract Number GS00K93AGS6403.

Circle Reader Service Number 2

OS/2 Developer

JANUARY/FEBRUARY 1994

THE MAGAZINE FOR ADVANCED SOFTWARE DEVELOPMENT



EDITOR'S COMMENTS

Profit, Programming, and Products 5



CORPORATE STUDY

Judges Find a New Way to Client/Serve Papers 6



PROGRAMMING INSIDER

Ins and Outs of Dynamic Link Libraries 12



DATABASE

Exploring the DB2/2 Application Development Process 18

Prototyping Database Applications with VX-REXX 30

Whatever Happened to OS/2's Database Manager? 36

Database Development Tools Buyers' Guide 82



OBJECT-ORIENTED PROGRAMMING

SOMObjects Developer Toolkit: Sharing SOM Objects with DSOM 42

Building a Notebook with IBM C Set ++ Objects 54



GUI

A List Box Replacement 66



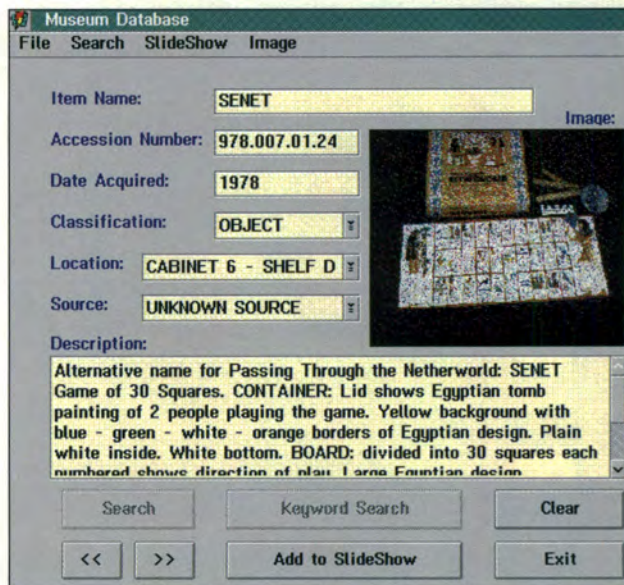
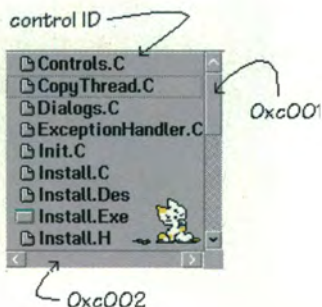
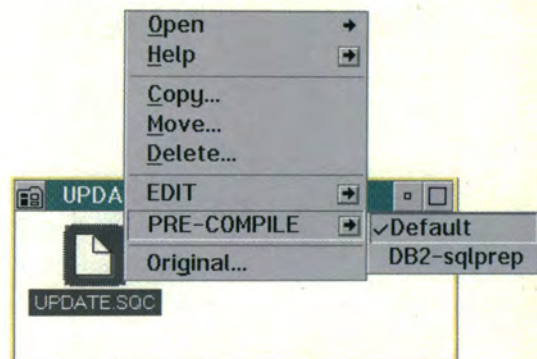
SOFTWARE TOOLS

Extending REXX with External Functions 73



PRODUCT WATCH

New Tools and Utilities 95





Programmer's Paradise®

Call for a
FREE
Catalog!



SourceLink v2.0 by One Up Corporation

SourceLink, the ultimate 32-bit programming development tool, combines the functionality of hyperlink source code access, a fully-functional editor and extensive file manipulation utilities into one very powerful toolset. Now you can find code, change code, spawn compiles and hyperlink directly to the source of error quickly and efficiently. Features point & click source code navigation and automatic generation of function call trees.



List: \$299 Ours: \$259
FAXcetera #: 6005-0003

Introducing the Lotus SmartSuite 32-Bit Bundle for OS/2! by Lotus Development Corp.

FREE
OS/2 2.1



Lotus' new SmartSuite for OS/2 is the only complete 32-bit solution for OS/2 desktops. For a limited time, when you buy SmartSuite for OS/2 or a SmartSuite for OS/2 Upgrade, you'll get OS/2 2.1 FREE. Act now to take advantage of this special introductory offer, while supplies last!

Upgrade: List: \$795 Ours: \$599
List: \$595 Ours: \$319
FAXcetera #: 4000-0027



Window Washer v2.0 by One Up Corporation

The latest version of the best-selling 32-bit screen saver for OS/2, with full system password security and the most complete monitor burn-in protection available today. Version 2.0 features many exciting animated effects, digital video and audio (plays CD's, MIDI or WAV files with program effects). Also utilizes TIF, GIF, BMP, & PCX backgrounds.

List: \$40 Ours: \$35
FAXcetera #: 6005-0001

SQL Objects++™

Database Class Library

SQL Objects++™ Database Class Library by Graphical Software Interfaces, Inc.

SQL Objects++™ Database Class Library is a powerful collection of object-oriented database classes that provide direct access to SQL and non-SQL databases. Each class is fully optimized to provide fast access to your data. No other access library is designed to fully utilize your OS/2 environment.

List: \$699 Ours: \$499
FAXcetera #: 1010-2601

Open Shutter v1.11 by One Up Corporation

Our easy-to-use screen capture process allows you to select any rectangular area, window, or entire desktop, and capture with a single user-defined keystroke or mouse click. Rotate, change colors, stretch/compress and then preview your modifications. Output to printer, clipboard, or soft copy in a variety of formats (BMP, ICO, TIFF, GIF, IMG, metafile, MacPaint).



List: \$70 Ours: \$59
FAXcetera #: 6005-0002

Ami Pro 3.0 for OS/2 by Lotus Development Corp.

OS/2 users now have a native, 32-bit version of the award-winning Ami Pro 3.0 for Windows. Written from the ground up for OS/2, the new version of Ami Pro provides the highest level of word processing, networking and development power among OS/2 word processors.



Upgrade: List: \$495 Ours: \$369
List: \$129 Ours: \$ 99
FAXcetera #: 4000-0011

WATCOM VX•REXX by WATCOM Int'l Corp.

WATCOM VX•REXX is an easy to use visual development environment for creating applications that leverage the capabilities of OS/2 2.x and exploit the Presentation Manager graphical user interface. VX•REXX combines a project management facility, visual designer and an interactive source-level debugger to deliver a very approachable and highly productive visual development environment.



List: \$199 Ours: \$99*
FAXcetera #: 1683-0016

*Pricing valid while supplies last.

Circle Reader Service Number 3

GUARANTEED BEST PRICES!(Call for details)
To order call: 800-445-7899

Corporate (CORSOFT): 800 422-6507

FAX: 908 389-9227

International: 908 389-9228

Customer Service: 389-9229

Programmer's Paradise Italia:

39-2-480-16053

For more information on the products
featured on this page call—

FAXcetera®: (201) 762-1378

Programmer's Paradise®

1163 Shrewsbury Avenue
Shrewsbury, NJ 07702

06JA4A

• All prices are subject to change without notice.
• Call for details on return policy and shipping charges.

9
9
8
7
5
4
4
0
0
8

EDITOR

Dick Conklin

EXECUTIVE EDITOR

Nicole Freeman

MANAGING EDITOR

Kelly A. Leighton

EDITORIAL ASSISTANT

Andrea Pucky

EDITORIAL ARTIST

Peter Altenberg

GROUP DIRECTOR

Don Pazour

PUBLISHER

Cathy Passage

ADVERTISING SALES

Holly Meintzer

(212) 626-2275

Michele Blake

(212) 626-2322

Dave Moreau

(212) 626-2318

Patricia Sutton

(212) 626-2498

MARKETING MANAGER

Susan McDonald

ART DIRECTOR/MARKETING

Christopher H. Clarke

ADVERTISING PRODUCTION**COORDINATOR**

Tom Marzella

DIRECTOR OF PRODUCTION

Andy Mickus

DIRECTOR OF CIRCULATION

Jerry Okabe

CIRCULATION DIRECTOR

John Rockwell

CIRCULATION MANAGER

Lucinda Formyduval

SUBSCRIPTION INQUIRIES

U.S.: (800) WANT-OS2

Foreign: (708) 647-5960



BY DICK CONKLIN

Editor's Comments

Proffit, Programming, and Products

OS/2 Developer continues to grow, adapting to the changing needs of our readers. As we begin another new year, we're adding three new features that I think you'll like. During OS/2's early years, we focused on the needs of independent software developers, as new commercial applications and tools took center stage. As more businesses downsize from mainframes to networked PCs, more corporate developers join the ranks of our readers. Sure, commercial ISVs and corporate programmers face many of the same technical challenges, but there are some important differences. The typical Fortune 500 programming department doesn't sell its applications through retail channels; it nevertheless must "sell" and satisfy very demanding in-house customers. ISVs creating standalone OS/2 products may not have to write client/server code, but they should be aware of the environment of their corporate users.

In this issue, we welcome back Brian Proffit, who graced these pages with a Software Tools column not long ago. Brian's new Corporate Study column will bring you real world examples of corporate OS/2 projects, from decision-making to user acceptance.

Dave Reich is also no stranger. Dave has written for OS/2 Developer, published books on OS/2 programming, and provided technical support to developers since Day One. His OS/2 Programming Insider column will deliver tips, techniques, hints, and examples only an OS/2 insider knows. Welcome!

BUYERS GUIDE

Not long ago we introduced a new Product Watch section, featuring news of new software tools, utili-

ties, and support for developers. In this issue we add a Buyers Guide—a listing of products related to the theme of this issue—OS/2 database development. This section is a "keeper" you'll want to refer to later. Thanks to our new managing editor, Kelly Leighton, and editorial assistant, Andrea Pucky, for making it happen—glad to have you with us!



Dick Conklin

WELL, WHAT DO YOU THINK?

Thanks to all of you who send us e-mail and participate in the OS/2 Developer section on

Contact	CompuServe	Internet
Dick Conklin (Editorial)	76711,1005, or OS2DF2 Forum	os2mag @vnet.ibm.com
Miller Freeman (Circulation)	71572,341	71572.341 @compuserve.com
Cathy Passage (Publisher)	71673,1163	71673.1163 @compuserve.com

CompuServe (forum OS2DF2). Your comments help us make the magazine better serve your needs. If you haven't written lately, drop us a line. If you have an idea for an article, pass it along. Better yet, if you have advice or a programming trick for your fellow OS/2 developers, write a short article about it. You don't need an English degree, just a knowledge of the subject. We'll meet you more than halfway. A short fax (415/905-2234) will start the process.

OS/2 Developer: Vol. 6, No. 1, January/February 1994

Subscription information: U.S.: One year (six issues), \$39.95. Canada, Mexico, and international surface mail, add \$16 per year for postage. Canadian GST no. 124513185. International air mail, add \$30 per year for postage. Foreign orders must be accompanied by payments in U.S. funds. For new orders and customer service, call (800) WANT-OS2 (800-926-8672) or (708) 647-5960 (fax: 708-647-0537). IBM employees and branch office customers can subscribe to OS/2 Developer through IBM Mechanicsburg's Systems Library Services (SLSS) using OS/2 Developer's order number: G362-0001. POSTMASTER: Please send address changes to OS/2 Developer, P.O. Box 1079, Skokie, IL 60076-8079. Allow eight weeks for subscriptions to begin. OS/2 Developer (ISSN 1073-0729) is published bimonthly by Miller Freeman Inc., 600 Harrison Street, San Francisco, CA 94107, (415) 905-2200. Application to mail at second class postage rates is pending at San Francisco, CA and additional mailing offices.

OS/2 Developer has applied for BPA membership.

© Copyright 1994 by Miller Freeman Inc. PRINTED IN THE U.S.A.



Corporate Study

*Just as they can write better programs by looking at sample code, developers can better design those applications by looking at real world problems and solutions. Our new Corporate Study column will examine the problems faced by organizations and how OS/2 products and tools helped to solve them. **By BRIAN PROFFIT***

Judges Find a New Way to Client/Serve Papers



Brian Proffit

An immense amount of information is processed behind the scenes in any county court system. Unlike a corporation, however, the users change frequently, bringing their own data processing preferences with them. Using OS/2 2.1 and CICS is one way to simplify the data exchange process and downsize it at the same time.

Frank Fitzsimmons faced this problem as director of Judicial Information Systems for Florida's 17th Judicial Circuit in Broward County. Broward, which contains Ft. Lauderdale, is the second-largest county with the second-largest judicial circuit in Florida. The judicial system (judges, state attorney, public defender, and sheriff) is composed of elected officials.

According to Fitzsimmons, "As parochial as they are, individuals count things differently, and therefore they have their own system. The adjudication of cases by a judge is counted differently than possibly [by] the state attorney or the clerk of the court. There are some standard reporting methods that we are responsible for at the state level, but internally, everyone does accounting slightly differently."

His organization ties hundreds of laptops together with numerous application servers and mainframe systems—whose data he doesn't own—into an integrated information system.

SIMILAR PROBLEMS

There are many similarities between the needs of Broward Judicial Information Systems (JIS) and a private sector corporation. "Government is a business, but people don't look at it that way," Fitzsimmons says. "With the 65 judges, I have 65 individ-

ual members of the Board, and the Court Administrator is the Chairman of the Board. It's my job to support their requirements and needs."

The information pyramid has three levels, with back-office systems at the bottom, management systems in the middle, and executive-decision systems at the top. For JIS, the back-office systems correspond to the civil and criminal case files handled by the clerk of the court. The middle includes jury management, probation management, and family-court management; much of this data is on IBM mainframes owned by other organizations. At the top are the programs that pull the data together into a meaningful form for the 65 judges and their judicial assistants.

When he first arrived, Fitzsimmons found an environment not suited for handling these needs. "When we first got here, they were using three different brands of PCs, all 8088 level, with 3270 PCs at the top end. Seven years ago, we put together a plan to migrate from where they were, which was nowhere, to where we'll be by the end of 1993."

Fitzsimmons's progress was limited by the tools available at the time, but technology caught up with his needs. "At the time, we put in DIS-OSS, which wasn't even called OfficeVision yet. We wrote a common front end on the 3090 host with CICS to give access to all the host-based applications and access to mail. Throughout the process, we have used mail to tie things together."

As he began to upgrade the desktop to PS/2 Model 55SXs, Fitzsimmons searched for an operating system. "We were in an environment where we had more than one application running simultaneously,"

LEARN OBJECTS FAST. DEVELOP 10x FASTER.

BusinessWeek

Smalltalk/V to the rescue. Simply by enabling large programs to be built from pretested software objects, it can provide tenfold leaps in programmer productivity and software quality. Perhaps more important, Smalltalk programs deal in terms programmers can understand.

Want to know the fastest way to learn objects – even for C++ developers? The experts recommend Smalltalk/V, the technology that's 100% pure objects.

FASTER, EASIER LEARNING.

With C++, you have to accomplish two huge tasks: learning object class libraries, and learning new C++ language syntax. Plus C++ is a hybrid of C with added object extensions. So odds are, you'll find yourself constantly falling back on familiar procedural methods and losing the benefit of objects.

Smalltalk/V has a simpler, more approachable language that lets you focus on learning objects instead of a new syntax. That's why thousands of professional programmers have found Smalltalk/V to be the fastest, most efficient way to learn the object-oriented paradigm. In fact,



C++ will make more sense once you learn OOP with Smalltalk/V. But we think you'll want to continue using Smalltalk/V for your application development.

DEVELOP 10x FASTER.

Greg Voss of *Windows Tech Journal* says: "It's not an exaggeration to say that most applications of significant size can be built ten times faster in Smalltalk/V than in C or C++." Gen Kiyooka of *Windows Tech Journal* agrees: "Nothing on earth can match the efficiency and productivity of experienced Smalltalk/V programmers. Nothing." And *Business Week* reminds us that some Wall Street firms now get their "computer models of

"You can work through a Smalltalk/V tutorial in a month and gain a better understanding of OOP than you'll get from six painful months with C++. What you learn will make C++ make sense."

— Greg Voss, OOP Alley
WINDOWS TECH JOURNAL

brand-new financial instruments done in days, not months."

EASY TO USE.

Here's how Smalltalk/V cuts development time by as much as 90%. •Easy-to-use integrated environ-

ment. •Incremental compilation shows immediate results of code changes. •Over 350 well-tested reusable classes built-in and 100's more available. •Class browsers and object inspectors to examine code and isolate errors. •Automatic memory management to eliminate memory related bugs. •Easy access to 3GL languages. •Integrated graphic debugger. •Available on Windows 3.1, Windows NT, OS/2 and Macintosh.

So if you want to learn objects fast – and develop 10x faster – punch this line of code into your telephone: (800) 531-2344 Department 706. We'll send you complete information.

SMALLTALK/V. 100% PURE OBJECTS.

DIGITAL

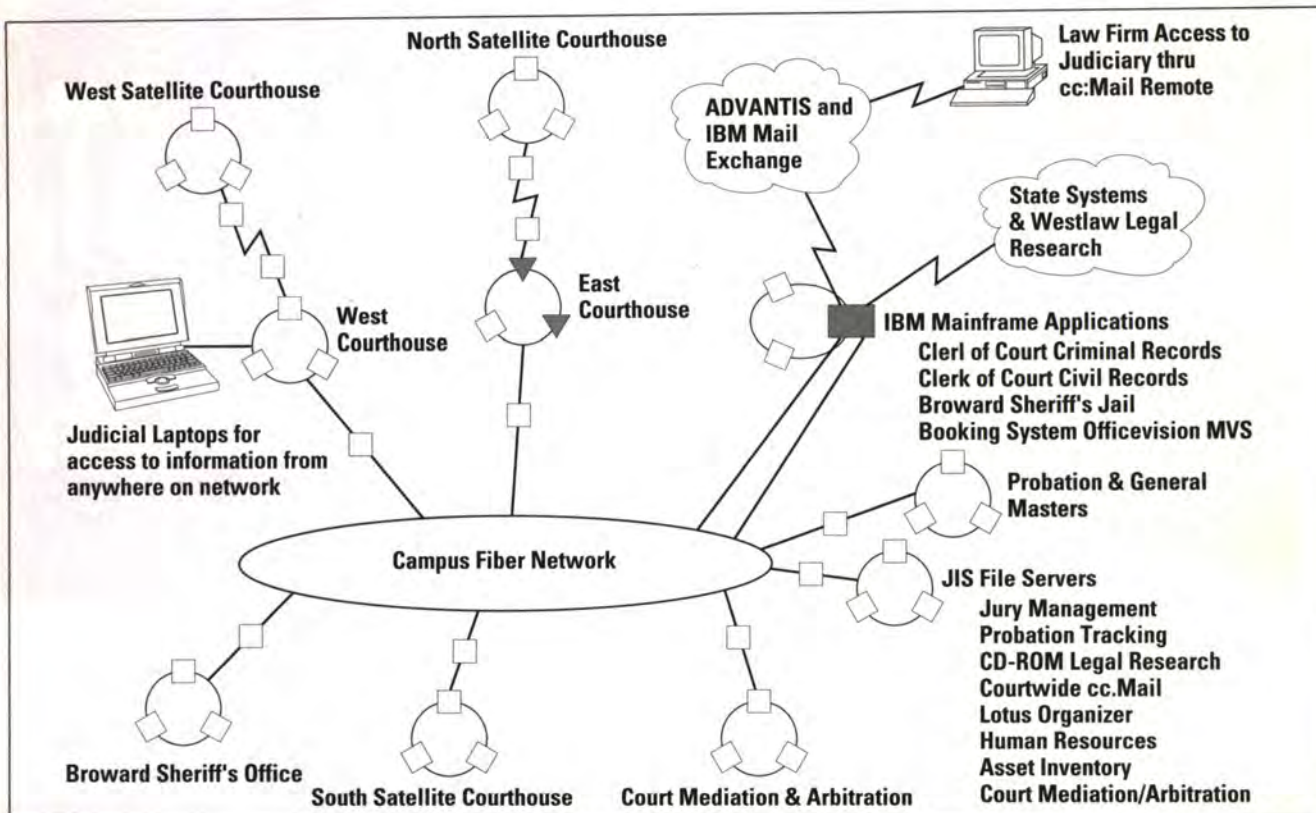


Figure 1. The 17th Judicial Circuit Court of Florida information network

Fitzsimmons says. For example, a judge or judicial assistant might often use two separate CICS regions and a word processor. They need cut-and-paste capabilities between the host sessions and word processor. They may also have multiple documents open, plus other Personal Information Manager (PIM) functions.

To support those needs, Fitzsimmons took an early look at OS/2. "We had the first version of OS/2 [1.0] in here, and we had a lot of problems. We went through beta releases and tested them. It was an extremely difficult year. We brought Windows in and looked at it. At the time, in the 286 and 386 days, Windows looked pretty good. It was an advancement."

Next, they simplified and standardized data access for the desktop systems. Like many organizations, JIS opted for Ethernet. "Our initial entry into a LAN-based system was a jury management system we did in Dataflex on the Ethernet," says Fitzsimmons. "We programmed it and redistributed it to several circuits throughout Florida, with the help of the state court sys-

tem. Immediately, we found that the Ethernet didn't have a lot of integrity. If you move a PC, the network comes crashing down." Changing the network at that point wasn't an easy task. As many MIS organizations must do battle with corporate offices to make changes, Fitzsimmons had to fight the county. He eventually received permission to shift from unshielded twisted pair to Type 1 twisted pair cable and token ring.

There was more to Fitzsimmons's seven-year plan. There were still many problems as a result of the data coming from several different places. "We needed to get to the data from the LAN, no matter if it was on a server, midrange, or mainframe. If a juror comes in late and the information is still up on the host, we need to be able to get that data down to the LAN right away."

JIS didn't own the data, however, and couldn't ask the county or state to change its way of doing business. So, court officials were running in batch mode at that point, shipping files from the host, working with the Dataflex pro-

gram on the LAN during the day, and batching updates back to the host.

On top of that, Windows wasn't all they had hoped for. "We were plagued by the dreaded unrecoverable application errors [UAEs]," Fitzsimmons reports. "Picture the system I described earlier: A judicial assistant is running two different host sessions with Attachmate to feed into a WordPerfect document and is in a hurry to get information to a judge. Suddenly, the assistant receives a UAE and has to reboot the system, restart the host sessions, and lose who-knows-how-much of the document. The first time that happened, the complaints began. The first time it happened while the judge was looking over the judicial assistant's shoulder, the situation became intolerable."

REASSESS AND UPGRADE

Fitzsimmons then began another round of hardware upgrades, moving to PS/2 56SLCs, and decided it was time to give OS/2 another look. "Our IBM account team came in here and convinced us to

Introducing
The Developer
Connection for OS/2

Development's

They're all here. All the tools, tips and techniques you need to send your OS/2® development rocketing up the charts. Brought to you by the original artists: IBM's own OS/2 development team.



Join The Developer Connection for OS/2,[™] and you'll receive the most timely and extensive information available to the OS/2 community.

Four times a year, you'll get a CD packed with the latest productivity tools, utilities and sample programs from IBM and others. A powerful browser and easy, graphical user interface let you locate any topic instantly in our comprehensive technical library.

Each CD includes the latest releases of smashes like "The Developer's Toolkit for OS/2," "Multimedia Presentation Manager/2 Toolkit," and "Pen for OS/2 Toolkit." Plus, you also get pre-release versions of many IBM products, operating systems, internal development tools, product demos, bit maps—you name it.

But wait, there's more. You also receive



greatest

The Developer Connection News, our newsletter filled with information about the latest OS/2 developments, new products, a Q&A column and much more. Plus, you get access to the Developer Connection forum on CompuServe[™],* where you can talk directly to the experts.

And here's music to your ears: Buy a year's subscription to The Developer Connection for OS/2 before February 15, 1994 for only \$199 and receive a free copy of the Clear & Simple[®] hit "Performance 2.1," a tuning kit for OS/2. Call 1 800 6DEVCON today, and start producing some hits of your own.

Operate at a higher level.[™]

hits.



*CompuServe membership is required. IBM and OS/2 are registered trademarks and The Developer Connection for OS/2 and "Operate at a higher level" are trademarks of International Business Machines Corporation. CompuServe is a trademark of CompuServe Incorporated. Clear & Simple is a registered trademark of Clear & Simple, Inc. ©1993 IBM Corp.



take a long hard look at CICS OS/2 to downsize applications on a midrange system we wanted to get rid of. When version 2.0 went golden and all the debug code was removed, it was a lot faster, but it was still pretty slow. It was more stable than Windows, though. One of the biggest factors was CICS OS/2. It allowed us to downsize some applications and streamline our access to host data for other applications. Today you'd call it client/server, but back then we just called it a distributed system, where we had data on both the host and the LAN, and we wanted to merge them to make the divisions more effective. Today, if that juror comes in late, with CICS OS/2 and distributed transaction processing, we can get that data to the LAN right away."

The same, distributed transaction technology solved another problem: monitoring probation. The probation-client tracking system is even more significant, because it is a revenue program. Clients pay a fee for the cost of supervision, so there is a lot of accounting in addition to the client tracking. If a person is adjudicated guilty and given probation, he or she needs to report to the probation department within 24 hours.

we need. That's a fully multithreaded application, and it took the combination of CICS and OS/2 to make it possible."

A PLATFORM FOR THE FUTURE

Now that the platform is in place, Fitzsimmons will exploit it further. "We're migrating from OfficeVision/MVS down to Lotus cc:mail. By the end of 1993, all judges and their judicial assistants will be running OS/2 2.1 with Communications Manager/2 access to the host and all the host-based applications, plus WordPerfect, cc:mail (we're using IBM LAN Gateway/2 to interface between cc:mail and OfficeVision), and an appointment calendar, probably Lotus Organizer."

Fitzsimmons believes there is a message to software developers in their choice of cc:mail and Organizer. "We looked at WordPerfect Office for three months before moving to cc:mail. Office provides better integration, but the kernel is still 16-bit and doesn't offer the performance we need. The 32-bit speed of cc:mail makes up for holes in the integration with Lotus Organizer." Speed is important; the system processes around 400,000 pieces of e-mail per month.

wherever we are with a GUI interface.

"Our next step is to extend our range outside the court and allow document transfer, probably with cc:mail Remote, to mailboxes in the local legal community. We want to give them the ability to exchange documents with the court, and we will be looking at adding imaging of documents to help. The good news is the platform to support this is in place. We've come full circle, from essentially nothing to extending beyond the courthouse and judicial system. This will make the judges and court system more responsible and effective in disseminating information."

ANALYZING THE RESULTS

Fitzsimmons glows about the changes at Broward County Judicial Information Systems. "The Service Pak for OS/2 2.0 fixed a lot of our problems. Version 2.1, especially the performance improvement, is super. We installed version 2.1 for general distribution as soon as it came out. We took old DOS 3270 PCs with a hodgepodge of software and replaced them with 56SLCs running OS/2 2.1, Communications Manager/2, and WordPerfect for Windows on a LAN server network. We went from a bicycle to a Cadillac. Key is that we had no problems with installation, implementation, or training."

Fitzsimmons believes that was partially due to the way they conducted the training, focusing on user benefits rather than operating platform features. "When we install, we do some basic training—a mouse, icons, and so on. These people have never dealt with these things. We show users how easy it is to pull several things together at once. They understand that it is easy to use, attractive, and most importantly, that they don't have to reboot the system frequently."

Brian Proffit has been in the computing industry for 20 years. He was part of IBM's OS/2 team but left to become the director of PC Week's Corporate Labs. He is now president of Visionary Research. Proffit is the author of OS/2 Application Development Tools, and OS/2 Inside & Out. He has written for OS/2 Magazine, PC Week, and Programmer's Paradise. He can be reached on CompuServe at 75300,1466.

Government is a business, but people don't look at it that way.

The records for clients are on a mainframe. To bypass the county's bureaucracy for adding or changing applications using the CICS system, Fitzsimmons again used Dataflex and CICS OS/2 to get to the data. Fitzsimmons says, "If a client hands you documents, the operator can run a distributed CICS transaction against the record on the host and bring it down. We can also now run (from a PC on the LAN) a report to list all the people that were supposed to show up. We can tickle records to find out if there is a violation of the probation terms. All of this was done manually, and paperwork used to fall through the cracks. Now, with CICS OS/2, we can put information in, bring it down, ship it out, and do whatever type of data entry or update

Having the network in place allows Fitzsimmons to provide additional services. "We have legal research CD-ROM systems running on the network. We're also implementing access to a remote paging facility and fax modems."

Fitzsimmons still needs to react to decisions made by the county and state. "There is going to be greater tracking of domestic violence cases, probably with an AS/400 application. Accessing that system from our network will further expand our range. We have a solid platform in token ring, we have a solid multifunctional operating system in OS/2 on the desktop, and with LAN Server and Communications Manager, we can access whatever we plug into the network from

Demand for break-through, 32-bit OS/2® applications is developing into a huge opportunity.

That's why we developed Borland® C++ for OS/2®—the easiest-to-use C++ development system for world-class OS/2 applications.

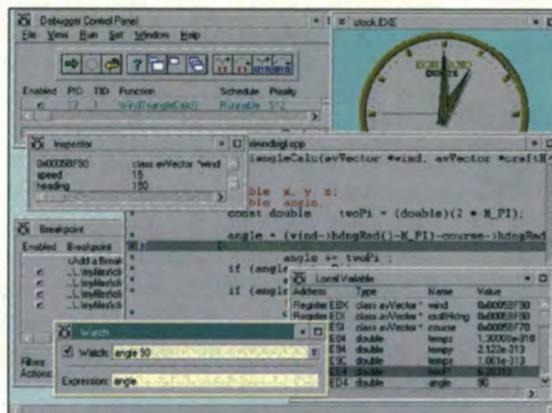
Borland C++ for OS/2 is a complete application development system. It combines an industry standard-setting Borland C++ compiler and

tools with the industrial-strength power of OS/2, so building C and C++ OS/2 applications is faster and easier than ever before. Borland's highly integrated graphical environment and visual tools raise productivity to a whole new level. The new, integrated Turbo Debugger GX finds bugs fast. And you can take full advantage of the full power of OS/2 with 32-bit code generation, pre-emptive multitasking and state-of-the-art optimizations.

To help speed your development along, Borland C++ gives you access to the largest resources of OS/2 libraries, controls and other add-ins. So if you're looking for fast, easy ways to create exciting, new, 32-bit OS/2 applications, get Borland C++ for OS/2

More exciting developments

from



Borland C++ for OS/2 has everything you need to succeed in creating leading-edge OS/2 applications.

today—and see what develops. To order or to find out more about OS/2 2.1 or Borland C++ for OS/2, call 1 800 3-IBM-OS2. In Canada, call 1 800 465-7999.

Operate at a higher level.™

Borland C++



OS/2's dynamic link libraries (DLLs) can be a big advantage—if they are used correctly. In this issue, our new Programming Insider column explains the tradeoffs, pitfalls, and benefits of DLLs. **BY DAVID REICH**

Ins and Outs of Dynamic Link Libraries



David Reich

Dynamic link libraries (DLLs) are an integral part of OS/2. Since OS/2 1.0, DLLs have been used to share code and reduce the system working set. While people are now accustomed to writing DLLs, I often see the concept of a DLL and how it should be used taken to the point of absurdity.

I have seen applications shipped with hundreds of DLL files. Others have one huge (as in the 5 megabyte range) DLL and a relatively small .EXE file. Still others have DLLs that are brought into memory when the application loads; a DLL stays there until the application terminates and is never called.

There may be good reasons for these application development design decisions. Having a large number of DLLs isn't necessarily wrong. However, developers are often unsure of why they are using DLLs, and many do not install or implement them as efficiently as possible.

WHAT IS A DLL?

A DLL is a file that contains a group of functions callable by many different processes. It is not an executable entity unto itself, yet after it is loaded it resides in memory like standard executable code. Architecturally, a DLL file is a cross between an executable program file and a set of functions, like the static C runtime libraries. The biggest difference is that a DLL does not have a `main()` routine. The DLL loads into memory and lives there.

A DLL is loaded by the loader as a result of a call from a thread of a process. This is done directly via `DosLoadModule` or indirectly by a program importing an entry point in the DLL. The system will set up the DLL in memory, which is then accessible to any

process in the system. It provides common routines to anyone in the system without forcing each application to have copies of the common routines built into each .EXE file. This improves performance and saves working set memory.

WHY USE A DLL?

Why should you put code in a DLL? DLLs reduce the size of the final, executable program and make code shareable. However, the primary functions of a DLL are to share routines among several processes, strategically delay code loading, and control the in-memory life of the code.

Used optimally, DLLs make your application perform better and increase the efficiency of the user's computer. Used incorrectly, they can slow down your application considerably as well as hurt overall system performance. Let's look at how each primary function fits into the scheme of things.

Sharing. First, a DLL is the tool for sharing code among processes or programs. Since DLLs are just a set of functions or library routines, a client process needs to know the parameter list for each of the functions it intends to call.

When writing a program, you simply call the functions in the DLLs by their name, and the DLL automatically links to the program. When you link a DLL to the application, a special external reference record is placed where the function call is. These references are also listed in the executable file's header.

Once the compilation is complete and the object file is generated, there is still an external reference to this function. In the link phase of application development, the DLL's LIB file is used to resolve the

For OS/2 2.x

UNLEASH THE POWER OF VISPRO/REXX™

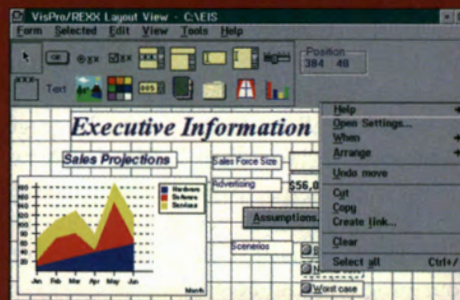


**Available
and Ready™**

for OS/2

VisPro/REXX is an easy-to-use visual programming environment that gives you the power to create your own OS/2 2.x GUI applications. Fast.

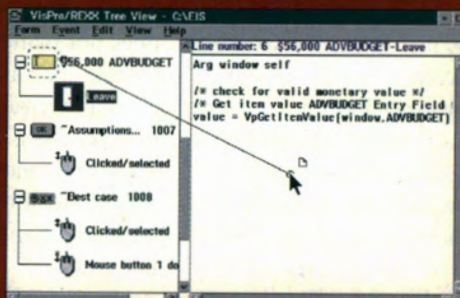
VisPro/REXX features the inherent advantages of Presentation Manager, a Workplace Shell based Project Manager, the Common User Access (CUA) '91 user interface and the SAA REXX programming language.



**Design
Your Layout**

VisPro/REXX offers easy access to OS/2 programming features, including buttons, lists, sliders, charts, business graphics, DB2/2, APPC and EHLLAPI. The upcoming SOM toolkit will allow you to add GUI objects.

Best of all, you don't have to be a REXX expert to use VisPro/REXX: the drag and drop programming feature creates the REXX statements for you.



**Drag and
Drop to
Build Your
Program**

With VisPro/REXX, you can quickly develop OS/2 CUA '91 applications, build client server programs and generate a single .EXE file of your masterpiece for royalty-free distribution.

UNLEASH YOUR POWER TODAY!

\$299

\$99 Bronze Edition

System Requirements:
OS/2 2.x, 5Mb memory and
1.5Mb hard disk space.

HockWare, Incorporated
P.O. Box 336, Cary, NC 27512-0336
Fax (919) 380-0757 Phone (919) 380-0616

HockWare™

Putting You in Control

VisPro/REXX is a trademark of HockWare, Incorporated.
OS/2, DB2/2, CUA, SAA and Workplace Shell are registered trademarks of International Business Machines Corp.
© 1993 HockWare, Incorporated. All rights reserved.

Circle Reader Service Number 7



function reference. Also, the reference in the `LIB` file of a DLL is not the code itself as it would be with, for instance, the statically-linked C runtime libraries. It is only a special external reference record pointing to the name of a DLL file and an ordinal (the number of the function within the DLL) within that file.

Loading. When a program loads, the loader reads through the `.EXE` header and finds all DLL references. If a DLL file is already loaded, the program resolves the references to it. If the DLL is not loaded, the program loads it and resolves the references. This usage scheme has its good points and bad points.

The scheme is good, because the application only needs to call the functions in the DLL by name and pass their parameters as if they were internal to the application. The application does not care where the functions are located. In this scheme, your programs can access important routines regardless of where they physically reside.

The drawback is that the DLLs get loaded when the application loads. If the functions referenced are never called, you loaded DLLs to resolve external references you never make. This wastes overhead. Also, for every external reference, the resolution, once made, is stored in application-swappable memory, increasing the working set of the application.

Another way to use functions in DLLs is to call the function by address rather than by name. This method saves the overhead of the resolutions at load time and saves application memory by delaying the loading of code until it is actually needed. This is done by calling `DosLoadModule` to load the DLL and obtain a handle to it, and `DosGetProcAddress` to obtain the address of the function. Once again, this method has strengths and weaknesses.

The strength is that you don't load code you aren't going to use. You also save space with respect to all the header resolutions you might otherwise generate. The weakness is that you will take a performance hit by loading the DLLs during application execution rather than having them ready when needed. Loading strategy is one of the most misunder-

stood points about DLLs.

Memory. Say you want to avoid the performance hit of loading the DLL during application execution. You want the library already loaded and ready. You want to call the function by name, and the header resolutions done. You don't care about the memory overhead of the resolutions, but want the DLL ready the instant you need it. You loaded the DLL when the program started.

Users will always use more memory than they have in the system. Therefore, the application will likely be running in a memory-constrained environment. Since memory is constrained and the user has not used the function yet, its page is likely to be paged out, so it has to be reloaded. Now the DLL has been loaded twice so the application can use it once. A solution is to call the function by address and only load it once.

SHRINKING YOUR EXECUTABLE

An example of using DLLs for an unintended purpose is to make the executable file smaller. All too often, I've seen DLLs implemented because the developer thought performance would be better if the `.EXE` file was smaller.

All code is loaded in 4K pages and accessed as needed. Whether the code resides in a DLL or in the executable does not matter. The pieces of the executable code are loaded as needed, so it is unnecessary to put code in a DLL just to make the `.EXE` smaller. By strategically placing functions in pages in the `.EXE` file, you will better overall system performance than if you just place functions in a DLL.

Another good reason for using DLLs is for Workplace Shell object classes. Because the current model of the OS/2 Workplace Shell is a single process, all objects must be accessible from the Workplace process. As such, object classes that you write will live in DLLs.

Use DLLs to create system extensions, strategically delay loading of code, and allow code sharing among processes. Workplace object class DLLs are also a system extension. Avoid DLLs that exist simply to make executable files smaller.

WHERE DO YOU PLACE DLLS?

There is no correct answer. There are architectural considerations when placing DLLs on a user's system and how (or if) to modify the `LIBPATH` when doing so. Since there is no one, best way to decide where to put them, understanding the choices based on the DLL's intended purpose will help you guide your users in placing these files correctly.

In the most generic case, a DLL is specific to an application or series of applications. Although a DLL's primary goal is to enable the sharing of code among processes, the majority of DLL implementations by applications are restricted to the single application or a series of related applications (as opposed to someone publishing a common set of system-extension routines, which is a great use for DLLs).

When the DLLs have such a specific and focused use, it's best to place them in the same directory as the executable file or files. Then, when installing the program and creating the program reference object for the Workplace Shell, you set the Working Directory setting to point to the directory containing the executable file.

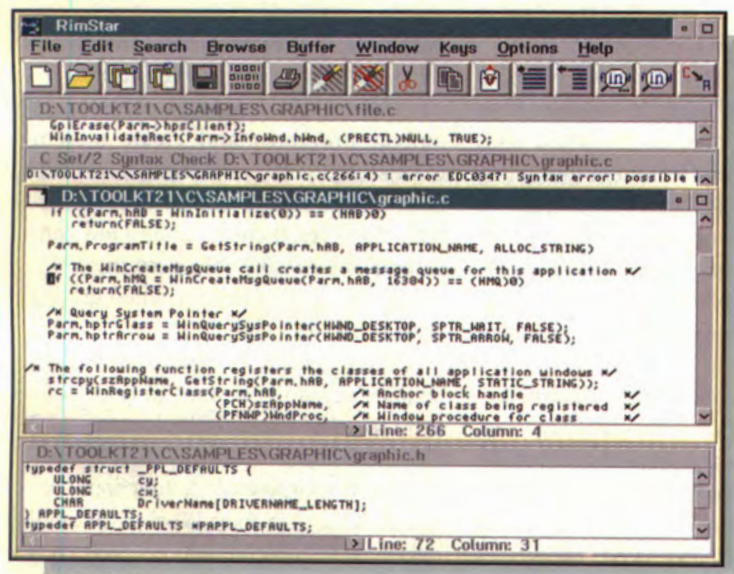
Starting with OS/2 2.0, the operating system installation program places a `.;` at the start of `LIBPATH`. Then, whenever the system is searched for DLLs and an explicit path is not specified, the system looks in the current working directory for the DLLs before looking anywhere else. If the working directory for your program is set to the directory where the DLLs live, they will be found instantly.

It is a bad idea to put your program's directory in the front of the `LIBPATH`. First, you will be moving the pieces of the operating system further back in the `LIBPATH`, slowing down the system. Your program's directory will be searched before operating system DLL files will be found. If every application did this, the operating system's as well as other applications' DLL path entries would be pushed back, causing a performance nightmare.

In the most general case, let the system find your DLLs through the `.;` in the `LIBPATH`, with you setting the working directory in the program reference object. You win, because the DLLs are

Brief's performance is over. Meet the new star.

The RimStar™ Programmer's Editor V2.0



If you're looking for a fully configurable, professional OS/2 PM programmer's editor to replace your current text-mode editor – we have what you have been looking for!

No hassle changing editors

We know changing editors can be a major hassle. You don't want to relearn a new set of keystrokes no matter how good the product. Well, you don't have to – we have Brief, Epsilon, Multi-Edit, SlickEdit, PWB and Borland IDE keyboard mapping waiting for you. If you're not using one of these, let us know and we will create the mappings you need.

Features designed to increase productivity

It has all the features you have come to expect, plus special features designed to anticipate your needs and increase your productivity. Features like a 'C' source browser that eliminates those time consuming searches for functions, global variables, typedefs, and macros by providing you with instant positioning to both definitions and references. Using our powerful 'C' macro language you can customize your programming environment to suit your individual needs.

All the features you need

- ✓ Complete 'C' macro language
- ✓ Auto-indent & template editing
- ✓ No limits on files, file size, or windows
- ✓ Bookmarks
- ✓ Line lengths to 16K
- ✓ Timed auto save
- ✓ Access PDK help for function under cursor
- ✓ Multi-threaded for no waiting
- ✓ Configurable tool bar
- ✓ Import/export to system clipboard
- ✓ Keystroke record/playback
- ✓ Block indent/outdent
- ✓ Brace matching
- ✓ Customizable menus
- ✓ Column, line and block selection, search and replace
- ✓ Integrates with Workframe/2
- ✓ Source browser for 'C'
- ✓ Unlimited undo and redo
- ✓ Multi-buffer regular expression search and replace
- ✓ Compile and jump to errors
- ✓ Complete on-line help
- ✓ Save and restore state between sessions
- ✓ OS/2 2.x 32 bit PM Multi-document interface

"My copy of Brief has been permanently retired. Keep up the good work!" - A.L.

Price \$299.00

Plus Shipping & Handling.

To order call (603) 778-2500.
90 day money-back guarantee.

RimStar Technology, Inc.

91 Halls Mill Road
Newfields, NH 03856-0938

All products and company names are trademarks or registered trademarks of their respective holders.

© 1993 RimStar Technology Inc.



found quickly, and the user wins, because you are not dangerously rearranging items in the LIBPATH.

Workplace shell class DLLs. Placing DLLs on the disk for Workplace Shell classes is an interesting and often religious discussion topic. There is no standard for it, and the architecture of the system places strict requirements on the functions in these DLLs.

When a new object class is created, it is registered with the Workplace Shell, which creates the appropriate entries in the operating system initialization files and desktop. The Workplace Shell needs the name of the DLL that contains the definition of the new object. If no path is specified, the LIBPATH is searched.

Let's say that you want to place the DLL with the executable files. This is not a bad idea, and it goes along well with the previous discussion. However, if the user remakes **INI** files, your class's registration will be lost. This is not a problem—just reregister it.

Let's say you have excellent foresight; when the program for the application runs, it checks to see if your class is registered, and if it's not, the application reregisters it. However, you don't know where the user put your DLL. You could assume that your DLL is in the same place as your program's executable or you could query to see where the executable lives.

Another solution is to put the DLL for the object class into a directory that always exists in the LIBPATH. The working directory won't cut it this time because you need to tell the shell where to find it, and the shell process' working directory is not the same as your executable's working directory. You could look for the boot drive and put it into, say, \OS2\DLL on that drive. But if the user reinstalls or moves the operating system to another drive without your DLL, you're back where you started.

You could also put your directory in the LIBPATH. (If you do decide to do this, remember to add it to the end of the LIBPATH.) This is not the best solution, either. Some users don't let application-installation programs change the CONFIG.SYS file. We let the program tell us what it wants, then add the things we

need and work around the rest.

For example, if a program tries to add something to the LIBPATH, I set the working directory for the program-reference object to whatever the program wants to add to the LIBPATH, rather than let it modify my LIBPATH. Although this works in a general case, it won't work for Workplace Shell object class DLLs.

By understanding how DLLs work, you can make the solution fit your exact needs. I prefer the first option: keep the DLL with the executable and pass the path to `WinRegisterObjectClass`. When the .EXE starts, make sure that the class is registered. If it isn't, register it.

Be aware that users may want to move your application to another drive or directory. If they do, the class DLL will not be found. Document that when moving the application to another drive, the user should reinstall it. This is also an important consideration for programs installed on LAN drives. Consider placing a copy of the class DLL on the client workstations.

You can also combine the first and second options. Copy the DLL to, say, \OS2\DLL. When the program starts, make sure the class is registered and the DLL is in \OS2\DLL on the boot drive (by using `DosQueryPathInfo`). If they aren't, take actions to fix the class registration. Deregister the class (even if it was not registered), copy the DLL from the application directory to \OS2\DLL, and register the class again.

The requirements of the current design of the Workplace Shell, combined with how DLLs are searched for and the things users do to the system, make choosing a location for your DLL very difficult. Use your knowledge of how DLLs work to make the best choice for your situation.

System-extension DLLs. System-extension DLLs need to be available to the operating system at all times, especially during system initialization. As such, they should always be in a directory accessible via LIBPATH. Since the processes accessing these classes are the system's processes, it is not feasible to put them in a working-directory entry.

Adding an entry to the LIBPATH is not the best solution; if you add it to the end,

all other directories have to be searched before the system-extension DLL is found. If you put the search entry at the beginning of LIBPATH, that directory is searched in all DLL searches.

I suggest you install system-extension DLLs in a directory that already exists in the LIBPATH, such as \OS2\DLL on the boot drive. The drawback: if the user reinstalls the operating system; in that case, your DLL will be erased. Then again, it should be obvious to the user that if he or she reinstalls the operating system, any system extensions must also be reinstalled.

SHOULD YOU EVEN USE A DLL?

DLLs used for their intended purpose are excellent tools. You can share code among programs and create system extensions, such as Workplace Shell classes, and make them part of the system. However, there are quite a few considerations to take into account when using DLLs. The first is whether you should use one at all.

Do you want to make your .EXE smaller? Or do you want to have your code accessible from multiple programs? Once you decide to use a DLL, look carefully at the purpose of the functions contained in it. Are they Workplace Shell object-class implementations? Are they just common routines for use in multiple programs? Are they system extensions? Depending on the answers, you have different considerations when deciding where to put them and how to modify the system configuration to use them.

Used appropriately, DLLs can make your application efficient and powerful. Used for the wrong reasons, they can slow down the users' system considerably. The choice is yours.

David Reich has been with the OS/2 development team in Boca Raton, Fla. since 1987. He has traveled the world giving seminars on OS/2 to users and developers, has written numerous articles for OS/2 Developer magazine, and is the author of *Designing OS/2 Applications* (Wiley and Sons Inc., 1993). Reich has an M.S. from the State University of New York at Albany. He can be reached on CompuServe at 76711,632 or on Internet at speedracer@vnet.ibm.com.

Client/Server 4GLs. Open as Pandora's Box.



n today's high technology marketplace, you hear a lot of stories about proprietary 4GLs for client/server application development. Stories that lure you ... that make 4GLs sound flexible, fast and open.

Beware the paradox: how can 4GLs be open when someone else holds the key?

Using client/server 4GL tools means opening Pandora's box of proprietary languages, vendor lock-in, and runtimes up to 16 times slower. With surprises like these, trusting your business-critical applications to 4GLs should scare you half to death.



With KASE:VIP, you get client/server solutions in standard languages like C, C++ and COBOL. Safe, secure solutions that leverage your current investment instead of replacing it. Using interactive visual design and code generation technology, KASE:VIP delivers the rapid development and increased productivity promised by 4GLs— but without unpleasant surprises.



Don't open Pandora's box. Get the facts in our free white paper, "No Strings, Boxes, or Borrowed Time: Overcoming 4GL Dependencies with Standard Language Client/Server Development Tools."

Just call 1 (800) 888-4335 or (404) 448-4240.

The leader in standard language-based client/server tools.

KASEWORKS™
△△△△ Enabling Client/Server Strategies

Fact. Not fantasy.



Database

This article describes the process for building DB2/2 database applications for OS/2 and how programmers can quickly develop application programs to meet their needs. **BY LUCIANO PEDRON**

Exploring the DB2/2 Application-Development Process

To manipulate data stored in relational database management systems (RDBMSs), application programmers need special tools and utilities. DB2 for OS/2 (DB2/2) 1.0 provides a third-generation language, application-development solution for building applications for DOS, Windows, and OS/2 clients.

This article describes the process for building DB2/2 database applications for OS/2 and how programmers can quickly develop application programs to meet their needs.



Luciano Pedron

APPLICATION PROGRAM ELEMENTS

Several basic elements are unique to DB2/2 application programs.

```
strcpy ( STATEMENT,  
        "UPDATE staff SET job = 'Clerk' where job = 'Mgr'");  
  
EXEC SQL EXECUTE IMMEDIATE :STATEMENT;  
if (SQLCODE != 0)  
    printf("\nUpdate Error:  SQLCODE = %1d", SQLCODE);
```

Figure 1. Embedding SQL statements

```
EXEC SQL BEGIN DECLARE SECTION;  
    char STATEMENT[81];  
EXEC SQL END DECLARE SECTION
```

Figure 2. Declaring host variables.

Embedded SQL Statements. SQL is embedded in a host-language source file. The SQL statements provide the database interface, while the host language provides the remaining support needed for the application to execute. Figure 1 shows embedded SQL statements.

SQL statements placed in an application program are not host-language-specific. In compiled host languages (C, COBOL, or FORTRAN) the precompiler converts embedded SQL into specific, language statements the host language can process. These specific, language statements are DB2/2 run-time API calls.

In REXX, SQL statements are passed to DB2/2 at run time through the **SQLEXEC** API. REXX applications are not precompiled. For additional information about SQL, consult the *DB2/2 SQL Reference Manual*.

Host Variables. Host variables are variables referenced by embedded, SQL statements. They transmit data between DB2/2 and the application. When you use a host variable in a SQL statement, you prefix the host variable with a colon (:). When you use a host variable in a host language, you omit the colon.

Host variables are declared in compiled host languages and are delimited by **BEGIN DECLARE SECTION** and **END DECLARE SECTION** statements. These statements enable the precompiler to find the declarations. See Figure 2 for declared host variables.

DB2/2 APIs. DB2/2 provides configuration, environment, utility, and additional APIs to include in application programs. For example, Figure 3 shows a call to the DB2/2 **INSTALL SIGNAL**



```
rc = sqleisig ( &sqlca );
```

Figure 3. Calling DB2/2 routines

```
EXEC SQL INCLUDE SQLCA;
```

Figure 4. Including the SQLCA structure

```
#include <sqlenv.h>
```

Figure 5. Including the SQLENV include file

HANDLER API (calling this API installs a signal handler to process a user-generated Ctrl+Break or Ctrl+C signal).

Two sets of DB2/2 APIs are provided for compiled languages: a set for C programs (SQLxxxx) and a generic set (SQLGxxxx) for COBOL and FORTRAN. There is no functional difference between the two sets; the generic set is provided for host languages that do not use null-terminated strings.

In REXX, DB2/2 APIs are not called directly. Instead, command-line versions of the APIs are passed as arguments to the SQLDBS API. For additional information about APIs, consult the *DB2/2 Programming Reference Manual*.

Data Structures. An application uses several data structures when communicating with DB2/2. The most commonly used data structure is the SQL Communication Area (SQLCA). DB2/2 transmits return codes through it to the database application. DB2/2 updates the SQLCA after most API calls and each time a SQL statement is processed.

The statement in Figure 4 causes the precompiler to insert into the host variable the definition of the SQLCA structure and a structure occurrence called sqlca. One method of defining and allocating an SQLCA structure is by embedding the SQL statement shown in Figure 4 in an application.

Include Files. Include files, provided with DB2/2, contain the necessary definitions of APIs, constants, data structures, and macros required by applications written in compiled host languages. They must be included in an application to use the defined DB2/2 APIs. Figure 5 shows

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sqlenv.h>                                (1)

EXEC SQL INCLUDE SQLCA;                            (2)

int main(void)
{
    EXEC SQL BEGIN DECLARE SECTION;
        char STATEMENT [80] = " ";                (3)
    EXEC SQL END DECLARE SECTION;

    SQL_API_RC rc = 0;
    printf( "\nSample C program: UPDATE");
    rc = sqleisig( &sqlca );                        (4)

    EXEC SQL CONNECT TO sample IN SHARE MODE;      (5)

    if ( SQLCODE != 0 ) {                            (6)
        printf( "\nConnect to error: SQLCODE = %ld", SQLCODE );
        exit(1);
    }

    strcpy( STATEMENT,                               (7)
        "UPDATE staff SET job = 'Clerk' WHERE job = 'Mgr' " );

    EXEC SQL EXECUTE IMMEDIATE :STATEMENT;          (8)
    if ( SQLCODE != 0 )
        printf( "\nUpdate error: SQLCODE = %ld", SQLCODE );
    else
        printf( "\nAll managers demoted to clerk!" );

    EXEC SQL ROLLBACK;                               (9)
    printf( "\nOn second thought...changes rolled back." );

    EXEC SQL CONNECT RESET;                          (10)
    if ( SQLCODE != 0 ) {
        printf( "\nDisconnect error: SQLCODE = %ld", SQLCODE );
        exit(1);
    }
    return 0;
}
```

Figure 6.a UPDATE.SQC sample program (see numbered explanation on page 20)

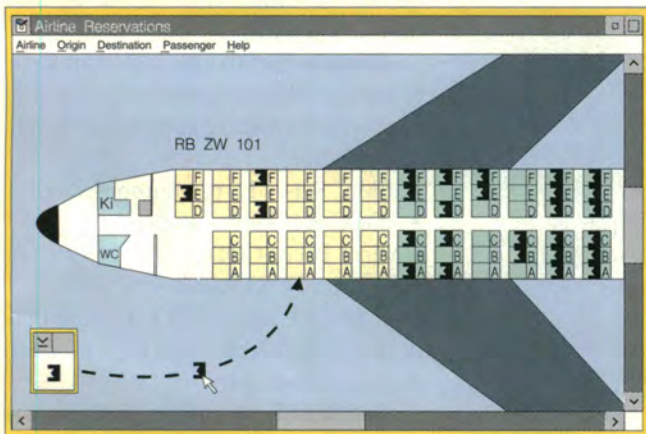


How the Sample Update Program Works

- 1. Define Database Manager APIs.** Applications written in compiled languages use include files to define DB2/2 APIs such as the `INSTALL SIGNAL HANDLER` API.
REXX applications do not call DB2/2 APIs directly. Instead, DB2/2 APIs are invoked using an intermediate API, `SQLDBS`. SQL statements are invoked similarly using the `SQLEEXEC` API. The `SQLDBS` and `SQLEEXEC` APIs must be registered using the REXX `rxfuncadd` function.
- 2. Define an SQLCA structure.** The `INCLUDE SQLCA` statement defines and declares an `SQLCA` structure and defines `SQLCODE` as an element within the structure. The `SQLCODE` field of the `SQLCA` structure is updated with error information by DB2/2 after execution of SQL statements and DB2/2 API calls.
REXX applications have one occurrence of an `SQLCA` structure, named `SQLCA`, predefined for application use. It can be referenced without application definition.
- 3. Declare host variables.** The `BEGIN DECLARE SECTION` and `END DECLARE SECTION` statements delimit the host variable declarations. Host variables are used to pass data to and from DB2/2. They are prefixed with a colon (`:`) when referenced in an SQL statement.
REXX applications do not need to declare host variables. Host variable data types and sizes are determined at run time when the variables are referenced.
- 4. Install signal handler.** Install a signal handler to intercept the `Ctrl+Break` and the `Ctrl+C` signals. Every production application should have a signal handler.
- 5. Connect to database.** The program connects to the sample database. The program requests shared access to the database. Other programs that attempt to connect to the same database in share mode are also granted access.
- 6. Check completion.** The `SQLCA` structure is checked for successful completion of the `CONNECT TO` statement. An `SQLCODE` value of 0 indicates that the connection was successful.
- 7. Copy SQL statement text to host variable.** The statement text is copied into the data area specified by the host variable, `STATEMENT`. Literal values in the statement are delimited with single quotes.
- 8. Execute the SQL statement.** The SQL statement is executed dynamically using the `EXECUTE IMMEDIATE` statement. There are two types of embedded SQL statements: dynamic SQL and static SQL. Dynamic SQL allows statement specification at run time, as in the example. Static SQL must be specified completely at precompile time.
- 9. End the transaction.** End the unit of work with a `ROLLBACK` statement. The result of the SQL statement executed previously can be either made permanent using the `COMMIT` statement or undone using the `ROLLBACK` statement. All SQL statements within the unit of work are affected.
- 10. Disconnect from database.** The program disconnects from the database by calling the `CONNECT RESET` statement. A pointer to the `SQLCA` structure is passed on the call. Upon return, the `SQLCA` is checked for successful completion.

This program contains scenes of a graphic nature.

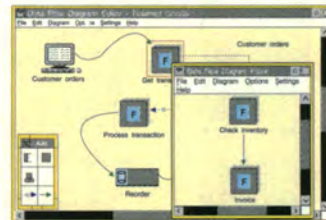
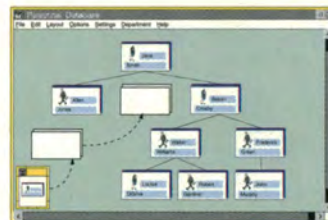
Introducing
Graphics
Interface
Kit/2



But don't worry. It's all in good clean fun. Because now there's a fast, cost-effective way to create graphic, easy-to-use interfaces for OS/2® 2.0 applications and tools. Introducing Graphics Interface Kit/2 (GIK/2), an AD/Cycle® product from IBM Programming Systems.

GIK/2 makes it easy to produce and manipulate symbolic representations of data without writing a single line of Presentation Manager® code. It automatically generates C code, where you can link the graphic symbols to the application data they represent. This not only speeds up your development cycle, it also helps you create applications that are intuitive and easy-to-use. Simplify transaction systems. Make databases easy to access. Bring organizational charts to life. Track inventory with simple icons. Any application can be made easier with the right graphic front-end. And GIK/2 can help you make it happen.

For orders or additional information, please call 1 800 IBM-CALL and ask for Department S71. If you need graphic proof, ask for our evaluation demo which, by the way, is rated G.



Database Application Programming Glossary

Structured Query Language (SQL). SQL is a standardized, database language. DB2/2 provides for data definition, retrieval, update, and control operations through SQL.

Application Programming Interface (API). An API is a published subroutine or function interface. APIs enable user-written programs to control database features (especially those not covered by SQL).

Host Language. A host language is a programming language in which database applications are written. API calls and SQL statements are placed in a host language source program.

Source File. A source file is a file containing SQL statements embedded within a host-language source program. A source file must have one of the following extensions:

- C source program—.SQC
- COBOL source program—.SQB
- FORTRAN source program—.SQF
- REXX source program—.CMD7

Embedded SQL. Embedded, SQL statements are SQL statements placed in a source file along with host-language statements.

Precompiler. A precompiler is a program that processes source files (for a compiled language) containing both host language statements and embedded, SQL statements. The precompiler converts the SQL statements to DB2/2 run-time services API calls and produces a modified source file with one of the following extensions:

- C program—.C
- COBOL program—.CBL
- FORTRAN program—.FOR

The modified source file can be processed by the host language compiler. The precompiler may also produce a bind file, a package, or both. Because REXX is an interpreted language, no precompiler or compiler is required.

Static SQL. Static SQL statements are statements that are prepared at precompile time.

Dynamic SQL. Dynamic SQL statements are statements that are prepared at run time.

Bind File. A bind file is a file that contains an internal representation of the SQL statements in a source file (and its nested EXEC SQL INCLUDE files). Bind files enable users to bind an executable program to a database without access to the source code.

Package. A package contains the same information as a bind file, but a package is stored in the database. Packages can be created at precompile time (by default) or explicitly with the binder. In OS/2 Database Manager (OS/2 Extended Edition and Extended Services), a package was known as a plan file.

Binder. A binder is a program that reads a bind file and stores the resulting package in a database. When an application is run, the package is used to access data in a database.

including the SQLENV include file.

Stack Requirements. Database applications require that you define stack space. In general, stand-alone applications require more stack space than client/server applications. The sample database applications provided with DB2/2 require a stack size of 32K.

SAMPLE UPDATE.SQC APPLICATION

Figure 6 shows the sample program UPDATE.SQC and demonstrates some of the concepts I've discussed. UPDATE.SQC accesses the staff table in the SAMPLE database and changes all managers to clerks. Then the program cancels the changes by performing a rollback.

APPLICATION ASSEMBLY

To build database applications, you need the following tools.

Text Editor. Use the OS/2 Enhanced Editor provided with OS/2 or any other text editor.

Compiler. Install your host language compiler. The supported list of OS/2 compilers for DB2/2 application development are:

- IBM C Set++ 1.0 (C component only)
- IBM C Set/2 1.0
- Micro Focus COBOL 3.0
- Watcom FORTRAN

If you are using REXX, be sure that it is installed from the OS/2 product disks or CD-ROM.

Database. Be sure that DB2/2 (single-user) or DB2/2 (client/server) is installed.

Sample Database. Install the sample database provided with DB2/2, if a database does not exist. Start DB2/2 and type SQLSAMPL G: at an OS/2 window. This will install the SQLSAMPL database on G:. If a drive letter is not specified, the database will be installed where DB2/2 is installed. When you attempt to connect to the database, you will be prompted for a user ID and password. The default user ID is USERID. The default password is PASSWORD.

Sample Programs. Install the sample programs that DB2/2 provides to assist database application development. The file SAMPLES.ZIP is stored in

Over 2,000,000
Installations Worldwide
The OS/2 software installer

InstallSHIELD™

New!
Version 2.0
Shipping for OS/2

The world's leading applications use it.

Shouldn't you?

First impressions last!

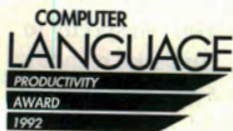
Use **InstallSHIELD** to create a bulletproof installation system for your application. Easy to use. Unconditionally guaranteed.

ENTIRELY NEW IN VERSION 2.0:

- Create Truly Professional Installations in a Matter of Hours
- Complete Logic/Flow Control Over Your Installation
- Intelligently Create Workplace Shell Objects, Folders, Templates and Modify .INI, CONFIG Files
- Fully CUA '91 Compliant, 32-Bit Multi-Threaded Execution
- Complete Support for Installing, Updating, Restoring, and Deleting Components of a Multi-Component Application
- Create Any Custom Dialog Within Our Script
- More Than 1,000 Pages of Documentation
- Over 200 New Features



Product and company names are trademarks of their respective holders.



Next Day
10 AM Delivery
Available

Stirling

Circle Reader Service Number 11

1-800-3 SHIELD

VOICE 1-708-307-9197 • FAX 1-708-307-9340 • COMPU SERVE: GO STIRLING

the root directory of Disk 1 of the DB2/2 product disks. To install the sample programs, insert Disk 1 into the A: drive and type the following code at an OS/2 Window:

```
pkunzip -d a:\samples\sqllib\samples
```

For the purposes of this article, the UPDATE.SQC application will be precompiled using the DB2/2 SQLPREP tool and compiled and linked using C Set++ (C component) in the WorkFrame/2 2.0 application-development environment.

The following compile options (flags) are only a small subset of the options provided by each compiler. In most cases, the default options are sufficient. Check your compiler's documentation for a complete description of available compile options. Suggested options for C Set++ are:

- **S**m allows migration extensions. This may help when migrating applications. For better portability consider **S2** or **Sa**.
- **G**m+ links with the multithreaded version of the library. Use this option if you want to write multithreaded applications.
- **C**+ performs compile only. This assumes that compiling and linking are separate steps.
- **O**- turns optimization off. It is easier to use a debugger with optimization off.

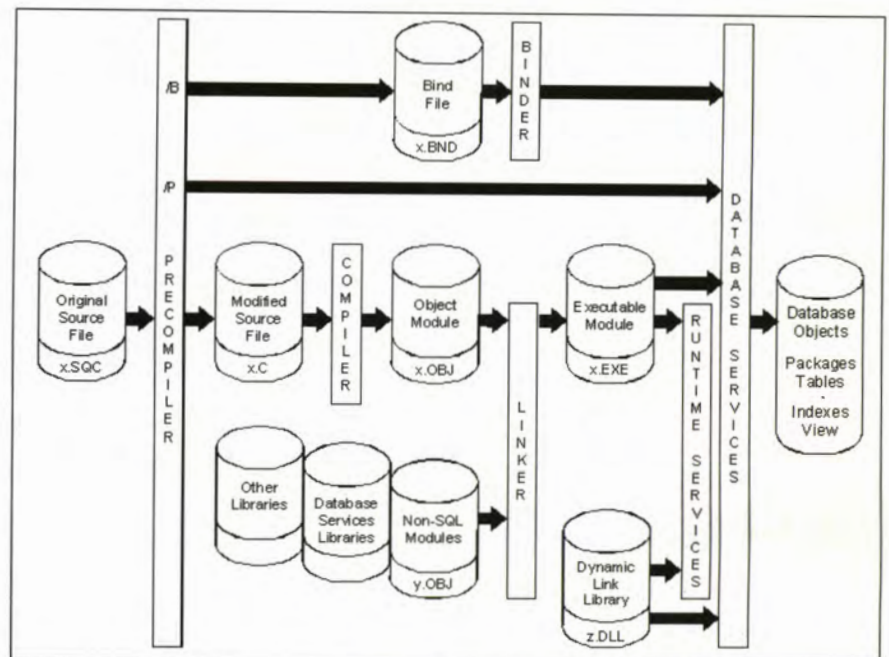


Figure 7. Seven basic steps for database application development

- **T**i+ generates debugger information.
 - **ST:32000** indicates the stack size may have to be increased more than the default.
- Other suggested linker options are:
- **DEBUG** includes debugging information.
 - **EXEPACK** packs executables.
 - **upm.lib** initializes user profile management (UPM). If using UPM APIs, be sure to link to the UPM Library.

- **ST:32000** indicates the stack size may have to be increased more than the default.
- Other suggested linker options are:
- **DEBUG** includes debugging information.
 - **EXEPACK** packs executables.
 - **upm.lib** initializes user profile management (UPM). If using UPM APIs, be sure to link to the UPM Library.

Database application development can be summarized in seven basic steps, as illustrated in Figure 7.

1. Create The Source Code. Write the source code in a standard ASCII file. Embed SQL statements and calls to APIs in the code. Save the file with the correct file extension, such as .SQC for programs written in C.

2. Precompile the Source File. Only precompile source files with embedded, SQL statements. SQLPREP converts SQL statements into comments and generates language-specific calls for the SQL statements. You can specify to build the package, bind file, or both. This step produces a modified source file, an optional bind a file, and an optional package.

3. Compile the Modified Source File.

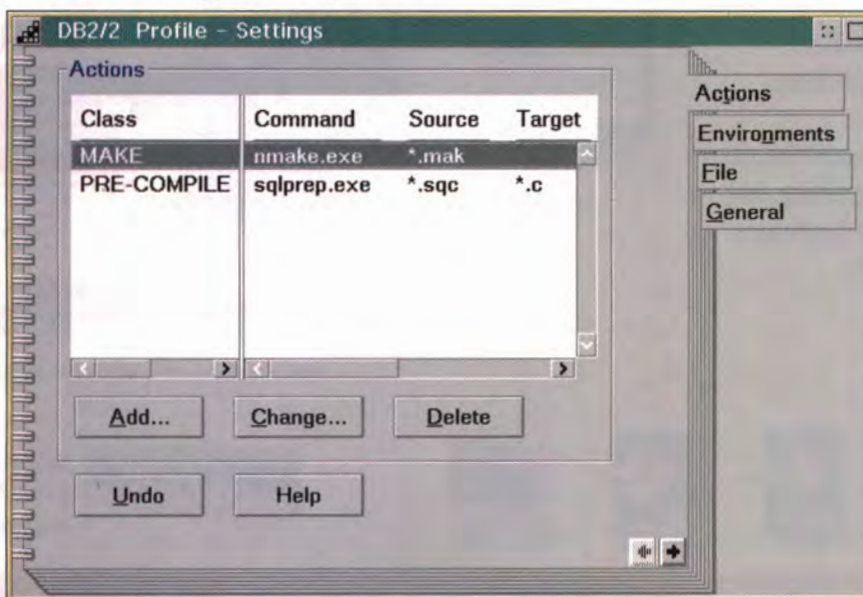
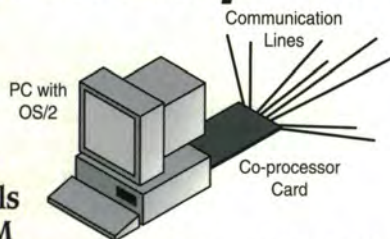


Figure 8. The settings notebook for a DB2/2 action profile

Cut your development time



Quadron's OS/2 development tools team up with IBM

ARTIC co-processor cards to give you flexibility, communication power, and development speed.

Use Quadron software and ARTIC cards to gain extra ports and enhance host PC performance by moving protocol processing to the cards. Develop standard and custom communications for async, BISYNC, HDLC, X.25, LAP-B, or your own special protocol. And if you need more than one protocol, no problem—just run them at the same time on the same card.

Programmers feel right at home with our software. Just program in standard C, and use our messaging API to link OS/2 threads with

tasks on the card. Then check it out with our symbolic debugger and real-time analysis tools in OS/2 windows.

Our easy-to-use software serves applications as diverse as telephone switching, financial transactions, satellite communications, and process control. If any of this sounds like a match for your needs, call us today.



209 East Victoria Street
Santa Barbara, CA 93101
Fax: 805-966-7630
805-966-6424



Trademarks are the property of their respective owners.

Circle Reader Service Number 13



The Impact Formatted Entryfields are a completely new window class which allows the developer to specify a format string when the entryfield is created. Formatting can be used for fields like Phone Number, Dates, Social Security Number, etc.

The Impact Formatted Entryfield dynamically formats itself as the user inputs data.

The Impact Formatted Entryfield support assures uniformity in data entry.

The developer has complete control to create custom fields like;

- Numeric entry only
- Alpha characters only
- Special delimiting characters
- Value ranges
- Currency support and, much more.

Easy to use! Just change the window class in your resource file.

Download free demo of Impact Entryfields from our BBS.
(818) 879-7405

Add Impact to your application



5889 Kanan Rd.
Suite 330
Agoura Hills
CA 91301

To order call or write
(800) 676-9390
(818) 879-5592
FAX (818) 879-5593

Circle Reader Service Number 14

SQL Objects ++ Database Class Library

Inherit The Power™

Database Access The Way It Was Intended. A collection of powerful, easy to use database classes that provide *complete* access to SQL and non-SQL databases:

Oracle	Sybase
SQL Server	Ingres
Informix	DB2/2
DDCS/2	Netware SQL
Watcom SQL	SQLBase
Btrieve	DB2/6000

The Object-Oriented

Approach. No longer are you restricted to the lowest common denominator factor when developing database applications. Stop using simple generic C APIs. Inherit the power *now*. Access database specific features with powerful C++ classes. No more embedded SQL or precompilers. Realize more flexibility than ever before.

Multi-Platform Support (16 and 32 bit). Whether you use OS/2, UNIX, DOS, Windows or NT, SQL Objects++ Database Class Library is designed to take advantage of your native environment. Most major C/C++ compilers supported.

NO RISK and NO ROYALTIES.

Use it for 30 days. If you're not satisfied, return it for a full refund. Order today. Prices begin at \$249.

**Competitive
Upgrade
\$249**

**GSI Graphical
Software
Interfaces, Inc.**
47 Stonewall Street Cartersville, GA 30120
Sales: (800) 876-6585
(404) 382-6585 fax (404) 382-6374

SQL Objects++ and Inherit The Power are trademarks of Graphical Software Interfaces, Inc. All company and product names are trademarks or registered trademarks of their respected owners.

Circle Reader Service Number 15

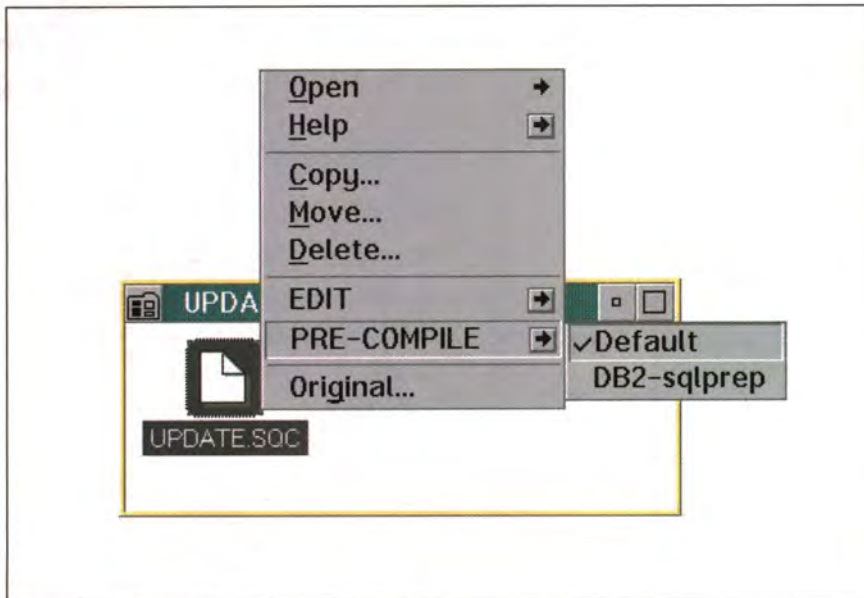


Figure 9. Performing a precompile using PRE-COMPILE

Compile all the modules, including the ones that were precompiled. This step produces object modules.

4. Link the Object Modules. Link the object modules with the proper link libraries to produce the executable file.

5. Bind the Application to a Database. Bind application to the databases to

create a package that contains instructions for accessing the data requested by the application. This step uses the bind file that was created in the precompile step. Binding is not necessary if it was done with precompilation.

6. Run the Executable File. Run the application and verify the results.

7. Distribute Executable and Bind Files.

Distribute the executable and bind files for your application, if your application is to be used with different databases or on different workstations. You will then need to bind the application to each database it will access.

PRODUCTIVITY USING WORKFRAME/2 v. 2.1

DB2/2 can be used with the new WorkFrame/2 2.1 application development environment to gain higher levels of productivity. The first step in integrating DB2/2 is to create an action profile. An action profile is a set of actions associated with one or more projects. Figure 8 shows the settings notebook for a DB2/2 action profile. The PRE-COMPILE action, which identifies the SQLPREP tool, has been created.

The PRE-COMPILE action performs a precompile of the application using SQLPREP and automatically binds the application to the sample database. PRE-COMPILE will also appear in the icon's pop-up menu. When you want to perform a precompile in the Workplace Shell, just point to the

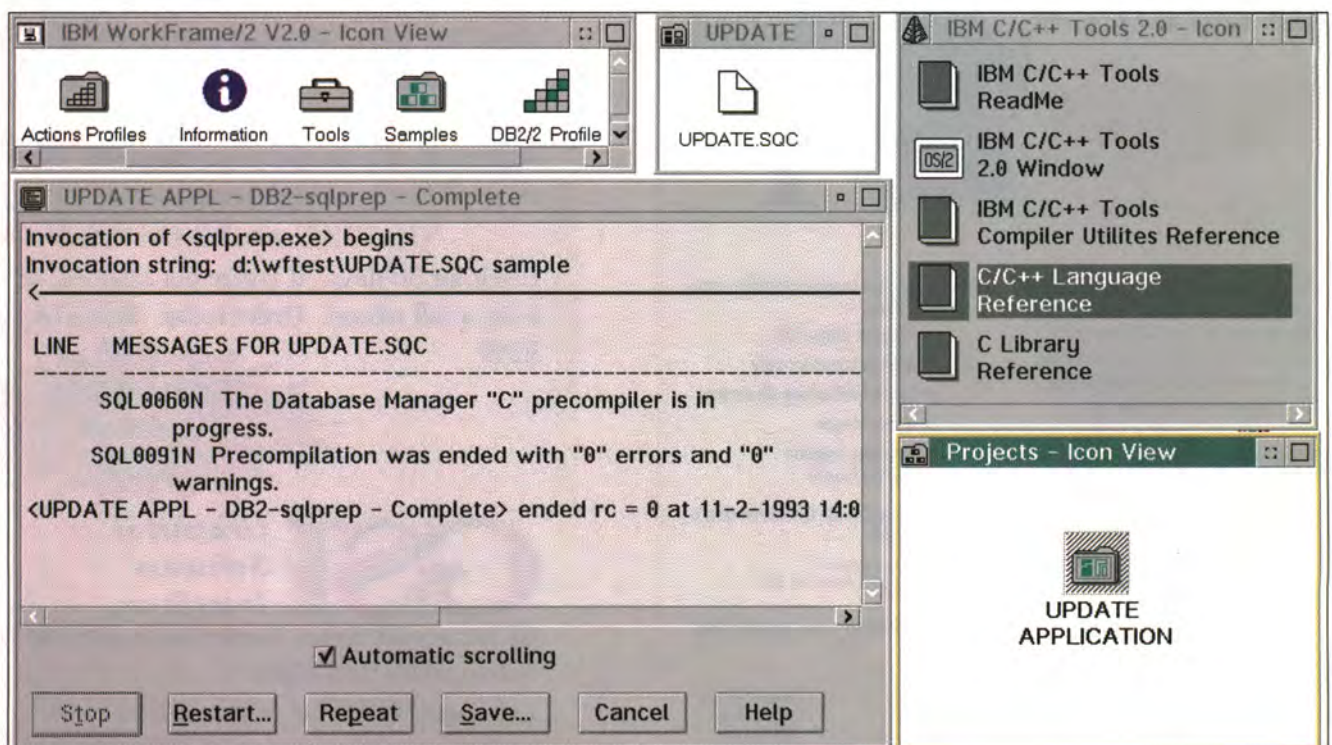
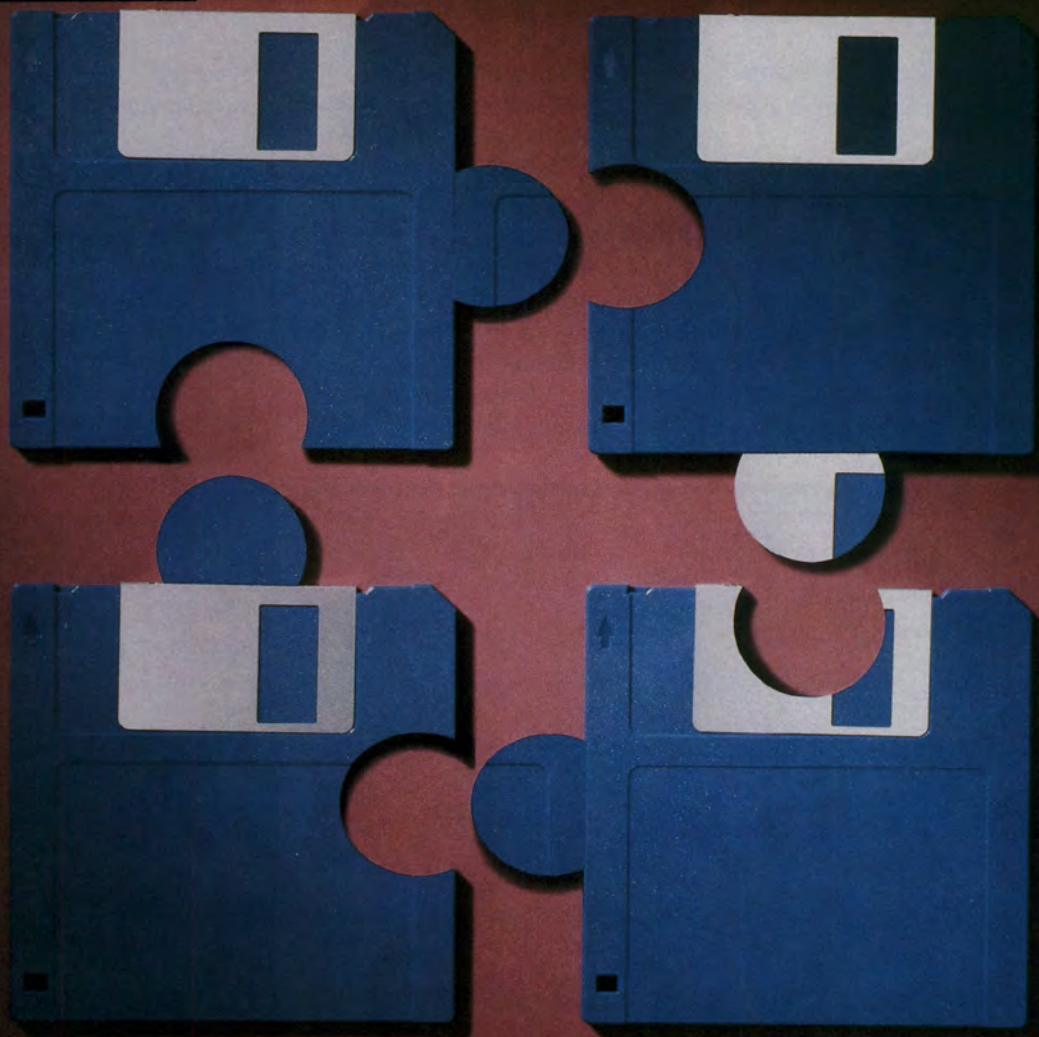


Figure 10. Displaying the output from the precompile

Relational Database Software

Client/Server



Applications Development

End User Tools

INGRES. The complete OS/2 solution.

Now available on OS/2 - The ASK INGRES Intelligent Database. The very same database that has proved so popular with blue chip organizations and high-end professional users in commerce, industry and Government.

Fully featured, fully portable, fully scalable.

The ASK INGRES Intelligent Database is a leading edge, intelligent, relational database, ideally suited to client/server applications. ASK INGRES supports all major desktop and UNIX networking standards as well as IBM SNA, enabling communication with DB2 and IMS.

The ASK INGRES Intelligent Database is complemented by a range of professional-standard rapid application development tools. ASK/Windows4GL - portable, object-oriented, fully featured and

GUI-based - combines intuitive user friendliness with unparalleled power and functionality. ASK/VisionPro packs equally impressive performance into character-based application development and includes such features as automatic code generation. Users can also access the database using ASK/Query and Reporting tools or tools from industry-leading software vendors. In addition, users have the pick of an ever-growing number of third-party ASK INGRES-based applications.

For a free INGRES OS/2 Solutions Kit, please call 1-(800)-446-4737 or 1-(510)-769-1400 (outside North America).



ASK Group



Circle Reader Service Number 16



UPDATE.SQC icon and select **PRE-COMPILE** from its menu, as shown in Figure 9.

You can choose to display the output from the precompile in a text window, a full-screen window, or a WorkFrame/2 monitor window, as shown in Figure 10.

The output from the **PRE-COMPILE** action is the file **UPDATE.C**. **UPDATE.C** can now be compiled using C Set++ or C Set/2 and linked to

create **UPDATE.EXE**.

For additional information about WorkFrame/2, consult the WorkFrame/2 2.1 Introduction manual.

CONCLUSION

DB2/2 provides a third-generation language application development solution that allows programmers to develop database applications. The WorkFrame/2 2.0 provides an application- development environment

designed for increased programmer productivity.

ACKNOWLEDGMENTS

The author would like to thank Grant Leclerc (DB2/2 Development) for his support. Input for this article was provided by the DB2/2 Toronto Information Development (TID) team. The author would like to acknowledge TID's work in providing excellent, user documentation for DB2/2. Lastly, the author would like to acknowledge Judy Wilson and the WorkFrame/2 Development team for their assistance.

Luciano Pedron is a product planner with DB2/2 and DB2/6000. He joined the IBM Toronto Laboratory in 1982, moving to the Image Systems Center in 1986. He has been working in database technology planning since March 1992. He received a B.S. in computer science from Lakehead University, Thunder Bay, Ont., Canada.

REFERENCES

- DB2/2 Information and Planning Guide* (IBM Doc No. S62G-3662)
- DB2/2 Guide* (IBM Doc No. S62G-3663)
- DB2/2 Programming Guide* (IBM Doc No. S62G-3665)
- DB2/2 Programming Reference* (IBM Doc No. S62G-3666)
- DB2/2 SQL Reference* (IBM Doc No. S62G-3667)
- WorkFrame/2 Version 2 Introduction* (IBM Doc No. S82G-3740)

OS/2 DEVELOPER BACK ISSUES

LIMITED NUMBER OF ISSUES AVAILABLE

The following Back issues of OS/2 DEVELOPER are available for \$9.95 each. (Add \$2.50 for shipping)

YES, PLEASE SEND ME:

- ☐ Winter 1990, DOS to OS/2 Conversion
- ☐ Fall 1990, Multimedia & Graphics
- ☐ Fall 1991, Computer Integrated Manufacturing
- ☒ Winter 1992 **SOLD OUT**
- ☐ Spring 1992, 32-bit Tools
- ☒ Summer 1992 **SOLD OUT**
- ☒ Fall 1992 **SOLD OUT**
- ☐ Winter 1993, Migration Strategies
- ☒ Spring 1993 **SOLD OUT**
- ☐ July/August 1993, Taligent Spotlight
- ☐ Sept/Oct 1993, Networking with OS/2
- ☐ Nov/Dec 1993, SOM/DSOM

Subscribe to OS/2 DEVELOPER for one year and save 33% off the cover price.

YES, PLEASE SEND ME:

- ☐ 1 year (six issues) - \$39.95 in the U.S.

Canada, Mexico, and international surface mail—add \$16 per year International air mail—add \$30 per year

Please check the appropriate boxes on this page and complete the form below, and mail this page to:

OS/2 DEVELOPER

P.O. Box 1079
Skokie, IL 60076-9772
or fax: 708-647-0537
phone: 800-926-8672 (800-WANT-OS2)

Please Print:

NAME _____

TITLE _____

COMPANY _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

☐ Check enclosed

☐ Charge my credit card:

☐ Visa

☐ Mastercard

☐ American Express

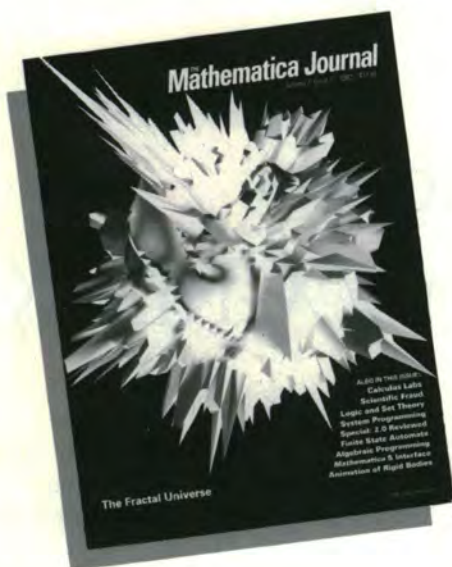
CARD NUMBER _____

EXPIRATION DATE _____

SIGNATURE _____

When it comes to scientific computing,
your best tool is *Mathematica* and...

THE **Mathematica® Journal**



Look Inside!

Each issue of *The Mathematica Journal* provides the latest hands-on technical information for all users. Regular columns include:

*** The Mathematica Programmer**

Showing you how to take advantage of Mathematica's powerful programming language.

*** From the E-Mail Bag**

Our editors comb the electronic newsgroups for tips, suggestions, and questions-and-answers to real-world Mathematica problems.

*** Mathematica 101**

Newcomers and experienced pros alike are welcome in this one-on-one classroom. Join us as we explore Mathematica's most exciting capabilities.

The Electronic Supplement

The Mathematica Journal is proud to announce The Electronic Supplement – a toolkit of Mathematica notebooks, packages, graphics, and utilities which optionally accompanies each issue. Currently, The Electronic Supplement is available on floppy disk for Macintosh and MS-DOS; more versions will be available shortly.

Don't Miss Out!

Each issue of *The Mathematica Journal* is filled with: book reviews; tutorial articles; news from Wolfram Research; calendar of upcoming conferences; full-color Mathematica graphics; reviews of complementary products; and more! Don't miss out, subscribe today!

YES, send my copy of *The Mathematica Journal* and enter my charter subscription for one year (three additional issues) or 2 years (eight issues), including the options I've checked below:

Options:

1 Year

- ☐ Journal only \$55
☐ Journal & electronic supplement (check format below) \$85
☐ Macintosh ☐ 3.5" MS-DOS/Windows

2 Years

- ☐ Journal only \$100
☐ Journal & electronic supplement (check format below) \$150
☐ Macintosh ☐ 3.5" MS-DOS/Windows

(Canadian/Mexican accounts add \$10; all other foreign accounts add \$32 for air mail. All funds in U.S. dollars. Please allow 6-8 weeks for delivery of first issue.)

Name _____

Title _____

Organization _____

Street _____

City _____ State _____

Zip _____ Country _____

Method of Payment:

☐ Check ☐ Bill me

☐ Visa ☐ MasterCard ☐ American Express

Card # _____

Exp. date _____

Signature _____

Mail your order to: *The Mathematica Journal*, P.O. Box 420096, Palm Coast, FL 32142-0096 U.S.A.

SAUR



*This article provides insight into how a visual REXX tool can be used with OS/2 and DB2/2 to create database applications quickly. Journey through the creation of the Games Museum database prototype, from database structure, to using SQL from VX-REXX, to storing images in an application. **BY GRACE WELCH***

Prototyping Database Applications with VX-REXX



Grace Welch

When the Museum and Archive of Games at the University of Waterloo, Canada considered downsizing its inventory database from the mainframe to a PC, it looked to two neighbors: IBM with its DB2/2 relational database system that was developed in nearby Toronto; and WATCOM, with its new VX-REXX visual development environment for OS/2 2.x that was created right in Waterloo. The two were good candidates for providing the solid, visual-development environment and powerful database platform the Museum needed. But rather than jumping into the development, the University decided to prototype the application with VX-REXX.

SYSTEM HISTORY OF THE MUSEUM OF GAMES

The Museum and Archive of Games manages the world's largest collection of games. The collection houses 5000 pieces, ranging from ancient card games to modern pinball. Twenty years ago, the museum boasted one of the most technologically advanced, collection-management systems in the industry. Each item in the museum was described in detail and recorded in a then-state-of-the-art VM/CMS-based hierarchical database called SPIRES. The conversion of the old, paper-bound ledger management system to an on-line, mainframe database greatly benefited the museum's curators. For a long time, the museum had wrestled with the problem of keeping information on each collection piece up-to-date. The museum operators needed to describe each piece as it was received, periodically adding information, such as a fading color problem and the resulting treat-

ment, to the description. The on-line database handled the accumulation of such information more conveniently than the paper system.

IMPROVING ON THE MAINFRAME SYSTEM

Two major factors caused the museum to seek a new, database solution. First, the nature of the museum's business changed. The museum shifted its focus from administration (or recording) to community and industry service. As a result, the database needed to change from an internal, administrative tool to a publicly available system. In particular, the museum wanted the database to provide easy-to-use, on-line information that included pictures of the exhibits. This way, customers could find information about games and their trends and history in a visual and interactive manner. It was not possible to achieve this ease of use within the bounds of the existing mainframe system.

Second, many university departments were downsizing their systems from the mainframe. During his search for an alternative platform, Elliott Avedon, a conservator of the museum, found a PC technology capable of meeting the museum's new requirements. Avedon viewed OS/2 and DB2/2 as powerful, user-friendly, and flexible building blocks for the Museum's database improvement. He started a project that examined the feasibility of moving the SPIRES database system to OS/2 and DB2/2.

MEETING THE CHALLENGE

Jennifer Uttley, senior researcher and developer of the University of Waterloo's Computer Systems



Group, took an interest in the museum database project. The project dealt with a problem in the area of her research: how to handle information about things, people, and events in a highly descriptive and usable manner. Within a couple of weeks, Uttley had created a working database prototype for the museum.

THE NEW LOOK OF THE GAMES MUSEUM DATABASE

The prototype database looked nothing like the SPIRES original. Instead of a text-based interface, the new database had a graphical user interface that included easily manipulated menus, drop-down list boxes, radio buttons, and push buttons. Figure 1 depicts the graphical user interface of the database prototype. The old, data-retrieval method of entering arcane SPIRES commands disappeared. Now, Museum customers could retrieve data by simply pointing and clicking on the new OS/2 interface. In addition, the database prototype offered new features, including picture storage, query by example, and ranked text searches.

THE GAMES MUSEUM DATABASE STRUCTURE

The Games Museum Database prototype is based on IBM DB2/2, a relational database that groups information into tables. These tables are related to each other through primary keys. The main table of the prototype consists of three types of fields. The fixed information fields include fields for the accession number (a unique identification number for each piece in the collection), the title of the item, the date that the item was acquired, and the market value of the item. The main table also includes a description field large enough to hold several paragraphs of information. Many smaller fields in the SPIRES database were collapsed to form this field. The new, all-inclusive description field offers an easier way to enter information that is not easily categorized. To ensure that the specific information can be easily found within the field, the prototype application includes the ability to retrieve records using a text search on the field. The main table also includes fields that contain the names of the images associated with each museum piece.

Other tables connect directly or indirectly with the main table of information. One of these tables includes categories (that is, object, photograph, or

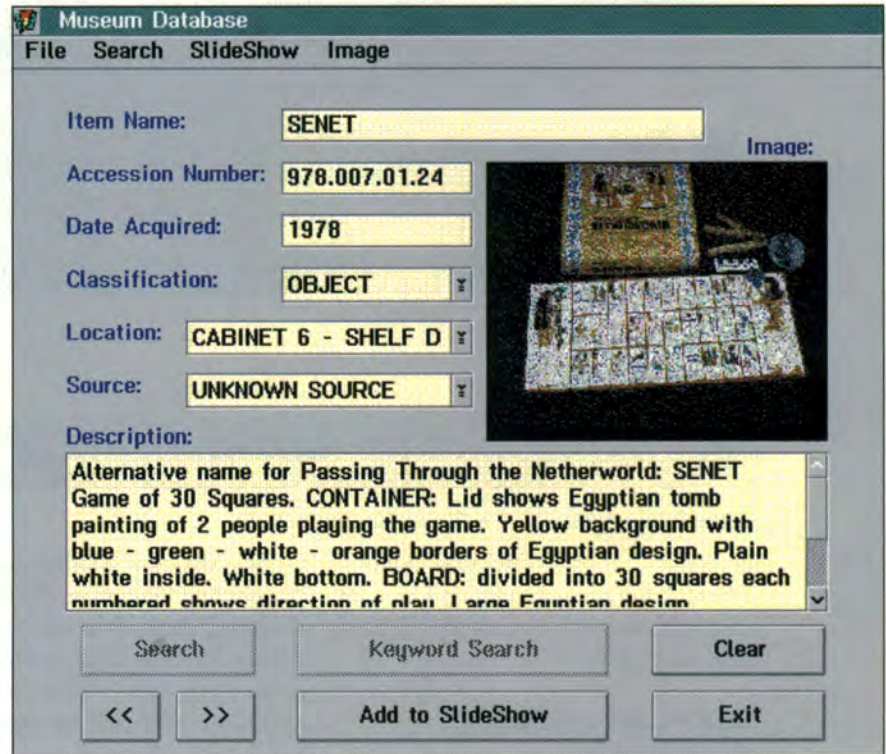


Figure 1. A view of the new Games Museum database application

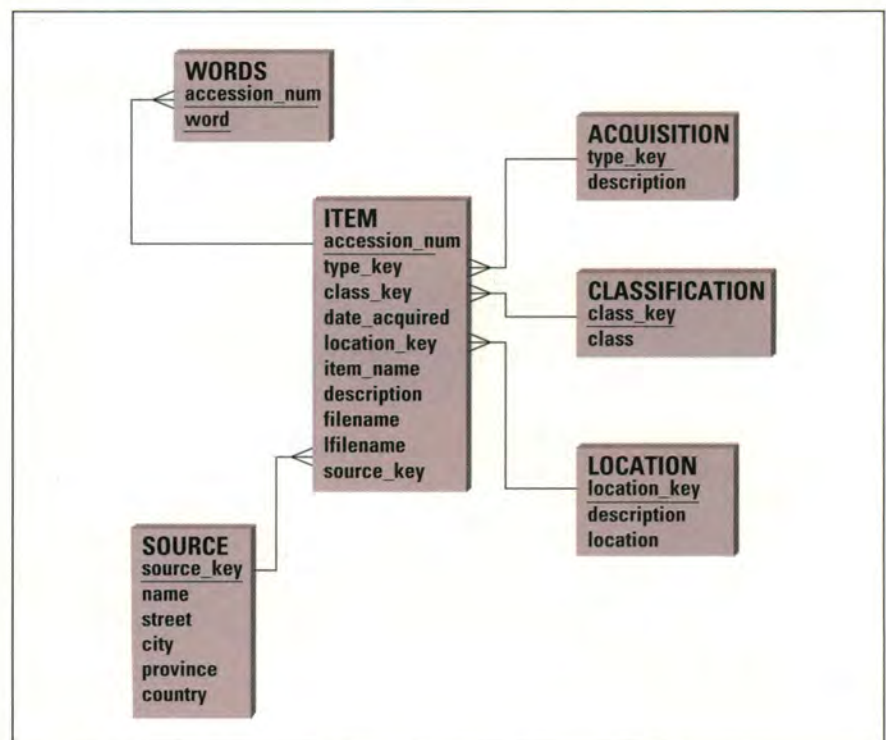


Figure 2. The database structure displayed in an entity-relationship diagram

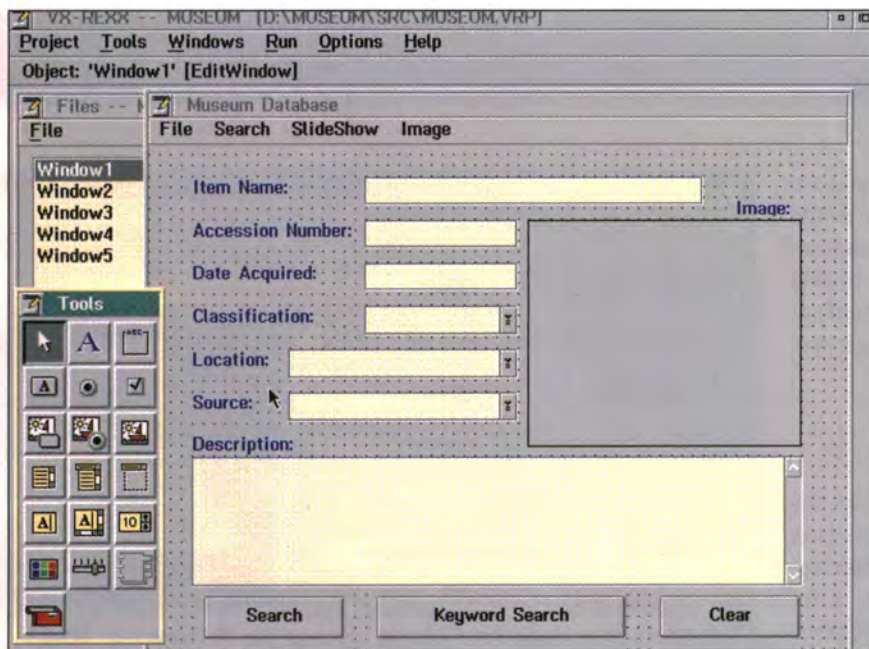


Figure 3. The main screen of the Games Museum database

book), and another includes donors. Figure 2 describes the Games Museum database structure in an entity-relationship diagram.

QUICK USER INTERFACE DEVELOPMENT

Developing a graphical user interface with a language such as C or C++ is complex. Using a visual tool, Uttley was able to create the graphical user interface for the Games Museum application quickly and easily. The VX-REXX development environment makes controls available in a toolbar. These controls were added to the

Games Museum database application by selecting them from the toolbar and dropping them onto application windows. Once placed on a window, each control's position, color, text, and other attributes were easily customized. Figure 3 shows the main window of the Games Museum database as it appears in the WATCOM VX-REXX visual development environment.

EVENT-DRIVEN PROGRAMMING

The user interface is connected to the application code by events. When a developer clicks on a button, for example, the button generates a click event.

Event code is written in standard OS/2 REXX. VX-REXX's drag-and-drop programming allows the developer to write the event routines without typing code. The developer simply drags and drops controls from the application window into the event routine, then selects the desired action from a list. Figure 4 illustrates this drag-and-drop programming feature. The code for each event can be edited separately, or all the code can be edited at once. Editing all of the code at once is useful for making global changes or for getting the big picture of an application. Developers also have the choice of using the built-in, VX-REXX editor or configuring the system to use an external editor.

CALLING DB2/2

DB2/2 includes an application programming interface (API) for REXX. An application, such as DB2/2, that has a REXX API provides routines a REXX program can call to interact with that application. Because VX-REXX is integrated with OS/2 REXX, it can be used with any REXX API. In addition, some applications are REXX-aware. Users can call macros written in REXX from within the application. The OS/2 Enhanced Editor and Lotus's AmiPro are two examples. VX-REXX can be used with this type of application to create visual macros.

Using DB2/2 involves:

1. Registering the DB2/2 routines
2. Starting DB2/2

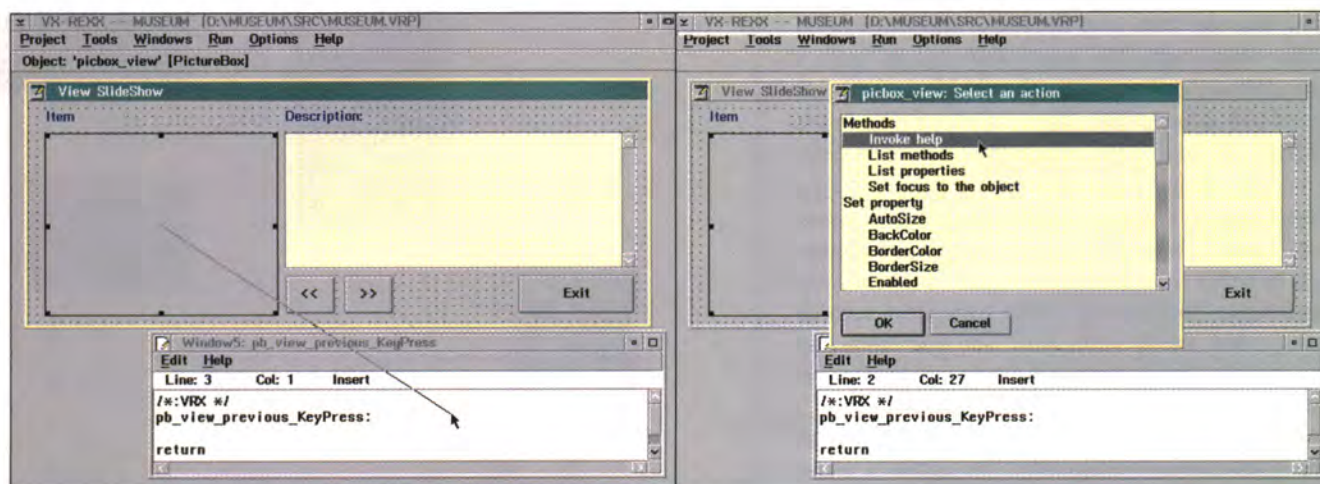


Figure 4. Drag-and-drop programming can eliminate the need to write code

Introducing IBM C Set ++ V2.1 and KASE:Set.

C Set ++ V2.1
including Kaseworks KASE:Set.
Starting at \$393 (U.S.) - CD-ROM Version.

Object oriented development.

C Set ++ Version 2.1 for OS/2® from IBM Software Solutions is one of the most complete object-oriented application development package you can buy.

Plus.

Its 32 bit C/C++ compiler, with its advanced code optimizer and

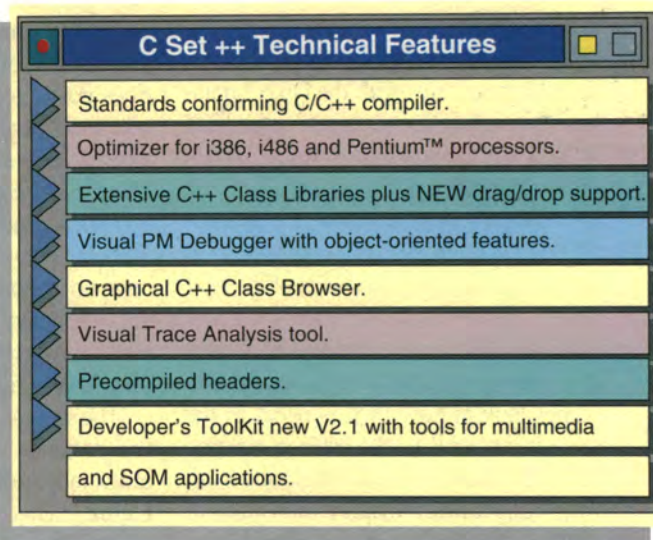
WorkFrame/2 V2.1 - a

completely new development environment based on

C Set ++ OS/2's object-oriented Workplace Shell - allows you to create up-to-the-minute mission critical applications.

You also receive a specially tailored copy of Kaseworks KASE:Set, a visual design and code generation tool which allows you to build visual applications even more quickly, with a minimal investment in learning time.

Plus!



To order C Set ++ for OS/2 including KASE:Set, or for further information call 1-800-342-6672 (U.S.A.) or 1-800-465-7999, ext. 676 (Canada). Or contact your local IBM software dealer.

IBM and OS/2 are registered trademarks and C Set ++ is a trademark of International Business Machines Corporation. KASE:Set is a trademark of Kaseworks. Pentium is a trademark of Intel Corporation. © 1993 IBM Corp.

Circle Reader Service Number 19

3. Declaring a cursor
4. Issuing a SQL statement
5. Displaying the data
6. Closing the cursor
7. Stopping DB2/2.

Many of these actions involve using two DB2/2 routines. `SQLEXEC` is used to process all SQL requests. `SQLDBS` is used to call DB2/2 environment and utility routines.

Issuing an SQL statement and displaying the data are described in the following section; more information about using DB2/2 with REXX is described in the *WATCOM VX-REXX for OS/2 Programmer's Guide and Reference* and in the *Database 2 OS/2 Programming Reference*.

USING SQL

The Games Museum database application communicates with the DB2/2 database through SQL statements embedded in REXX functions. For example, when the right arrow button is pushed on the application's main window, the embedded SQL statement, `FETCH`, is issued to retrieve and display the fields of the next record in the database. In Figure 5, a code sample from the click event routine of the `Next` button provides a behind-the-scenes look at how SQL is used. In the sample, the call to the `SQLExec` routine finds the next record. The following three calls to `VRSet` display the values of the three variables that were found.

CREATING A RANKED TEXT SEARCH

The Games Museum database application features a ranked text search on the description fields of each record. The ranked text search, called a "keyword search" in the application, requests a word or series of words from

```

/*:VRX*/
PBM_Click:

    call SQLExec 'FETCH C1 INTO:ITEM_NAME, :ACCESSION_NUM, :DATE_ACQUIRED'
    if SQLCA.SQLCODE \=0 then do
        call VRSet 'PBN', 'Enabled', 0
    end

    call VRSet 'EF_Item', 'Value', ITEM_NAME
    call VRSet 'EF_Accession', 'Value', ACCESSION_NUM
    call VRSet 'EF_Date', 'Value', DATE_ACQUIRED

    return

```

Figure 5. Code sample from the click event

the user to form a search pattern. The resulting matches are ranked by how well they match the original search words. The order of the search results does not reflect the order in which the words appear in the description; the results are arranged in descending order by the number of words in the text string that satisfy the search query.

Before the search, certain words that have no important meaning of their own are eliminated, such as *it*, *or*, *and*, *the*, and *that*. Characters such as brackets and punctuation marks are also removed.

The search runs very quickly, because the database includes a search word index table called "Words," as shown in Figure 6. This index contains the description text broken into individual words. Stop words and unnecessary characters are not included. A separate, VX-REXX program compiles the word index as part of the update procedure.

Suppose you want to search the Games Museum database for the string "An early American game board." The search procedure would issue the SQL statement in Figure 6. The `SELECT` state-

ment causes the database to return three fields: the accession number, item name, and number of words matched. Only descriptions containing at least one word from the search list will be selected. The `ORDER BY` clause causes the rows to be returned from the database in decreasing order of number of matches, thus ranking the matches.

TAPPING INTO OS/2 MULTIMEDIA CAPABILITIES

The Games Museum database application stores full-color images of museum pieces for many of the museum records. Like the Games Museum database application, other applications can benefit from this capability. Since a picture is worth a thousand words, pictures can replace descriptive text using only one field. They also accurately reflect the attributes of items and aid the usability of a database. Finally, users can use pictures for presentations and other external uses. For example, the Games Museum database application provides a slide show feature. The slide show consists of one or more item pictures and their attached descriptions. The slide show can be loaded on a disk and sent to customer sites to provide education and information. The slide show presentation is quick to develop, paperless, easy-to-use, and inexpensive. It can reduce the need for personal, on-site visits.

The museum pictures are stored in the database as digitized images. To create

```

SELECT words.accession_num, COUNT(DISTINCT Word), item.item_name
FROM words, item
Where Word IN ('EARLY', 'AMERICAN', 'GAME', 'BOARD')
    and words.accession_num = item.accession_num
GROUP BY words.accession_num, item.item_name
ORDER BY 2 DESC

```

Figure 6. SQL statement issued by search procedure

them, 35mm slides were scanned by a Nikon LS3510/AF. The resulting high-resolution files were converted to 256-color bitmap files using Image Alchemy, a conversion program. The program Wingif was then used to dither the files into a 16-color format that would allow low-resolution monitors to display the images.

The VX-REXX picture-box control was used to draw a box that would contain the images. At run time, assigning the path of the bitmap file to the picture box's `PicturePath` property causes the picture to be displayed.

Avedon also sees the potential of tapping into further multimedia capabilities

available from WATCOM VX-REXX. The Museum carries many games that involve sound and motion. To accurately depict these games and how they are played, the database could be modified to support audio and motion picture fields. In fact, these multimedia records could play on separate, simultaneous threads for added convenience and interaction.

THE FUTURE OF THE GAMES MUSEUM DATABASE PROTOTYPE

Avedon expressed enthusiasm for the future of the Games Museum database after reviewing Jennifer Uttley's Games Museum database prototype. The proto-

type allows him to present the proposed, database alternative to colleagues in a visual and realistic manner.

As for the final production version, issues regarding the transition of a hierarchical database to a relational database remain in debate. However, the ability to store images, a GUI front end, easy maintenance of the information, sophisticated, word search features, and usability of the database were all major improvements over the mainframe version of the database.

ACKNOWLEDGMENTS

I would like to thank the people who contributed to and reviewed this article, including Jennifer Uttley, Elliott Avedon, Rob Vietch, and Terry Stepien.

Grace Welch is a marketing specialist for Watcom, Waterloo, Ont., Canada, where she works closely with the C, C++, FORTRAN 77, Watcp, SQL, and VX-REXX product lines for OS/2. She recently received a B.A. from the University of Waterloo.

REFERENCES

Database 2 OS/2 Programming Reference, International Business Machines Corporation, First Edition, March 1993.

WATCOM VX-REXX for OS/2 Programmer's Guide and Reference, WATCOM International Corporation, 1993.

The Path To A Good GUI Isn't Always Clear

Navigate the maze with the NEW 2.1 Professional Developers ToolKit

Promises are cheap, GUI development tools aren't. Gpf 2.1 demo software is **FREE**. Try Gpf before you buy any tool and you'll agree that **Gpf 2.1 DELIVERS**.

With this powerful point and click visual programming environment you can create a CUA '91 Graphical User Interface for OS/2 Presentation Manager® or MS DOS/Windows® in as little as 10% of the time required to hand code the same design.

What You See Is What You Get programming reduces weeks of coding to hours. Quickly prototype and test your interface, then let **Gpf write the code**. Gpf generates error free, structured Object Oriented C or C++ source, complete with embedded SQL for OS/2 DataBase. Custom Help and Logic are added to controls as they are drawn to create full **Client/Server** or **Stand-alone** applications! Gpf is hosted on OS/2 2.x and generates **native code** for OS/2 2.x, OS/2 1.3.x and MS Windows 3.0 or 3.1 Of course this means maximum performance with **no run time module** and **no royalties!**

Seeing Gpf is believing!

- Design 32 or 16 bit GUIs for IBM OS/2® or Microsoft Windows®
- Be creative, WYSIWYG design environment
- Full application and/or DLL generation
- Minimal time to application delivery
- Full CUA '91 Control set, 32 or 16 bit
- Reusable Objects
- Automatic documentation

Circle Reader Service Number 20

Gpf

Try us, **FREE** Demo Software Available
Order Gpf Pro ToolKit today for just \$1440⁰⁰
Separately: Gpf 2.1 for \$1295⁰⁰ & GpfTools for \$195⁰⁰
Or, try: Gpf 2.0 Single Platform, now available for just \$495⁰⁰

Call Gpf Systems Inc. at: (800) 831-0017



* GUI Programming Facility



SYSTEMS, INC. Phone (203) 873-3300 • fax (203) 873-3302 30 Falls Road, Moodus, CT 06469



BY LEON KATSNELSON

Whatever Happened to OS/2's Database Manager?



Leon Katsnelson

Ever since PCs moved from tape cassettes and floppies to hard disks for storage, programmers have been seeking better ways to store and access data. Their quest for fast and efficient, data-access methods led to the introduction of such popular PC database systems as Borland's dBASE and later, Paradox. These products offered not only data access but also tailored programming languages (such as xBASE and PAL) and development environments for both novice and experienced, application developers. Over the years, dBASE and Paradox have become de facto database formats for standalone, PC applications.

The introduction of record locking and file sharing in DOS and the growing popularity of local area networks (LANs) caused many programmers to use LAN file servers for data shared by several users. While these innovations provided many benefits, they also had their shortcomings. File-server database management systems, such as dBASE and Paradox and Microsoft's Fox-Pro and Access, which send data from the server to a client PC for processing, were simply not efficient enough to handle the demands of mission-critical applications. Worse, they lacked guaranteed, referential integrity and transaction-processing capabilities. Nevertheless, these PC databases became a popular way to share application-specific data among a small number of LAN users.

THE MAINFRAME DATABASE

While PC programmers busily dealt with the complexities of robust, data-access mechanisms, mainframe programmers at IBM's Santa Teresa Labora-

tory implemented a new model for data organization and access known as a relational database. The result of this effort was a relational database management system (RDBMS); IBM Database 2 (DB2) was born. Later, IBM delivered an RDBMS for the VM and DOS/VSE operating systems (SQL/DS) and included RDBMS function in the OS/400 operating system for the AS/400 family of minicomputers.

In 1987, IBM and Microsoft jointly announced the arrival of a new generation of operating system, OS/2. Availability of a relational database was vital for OS/2 acceptance within the business community; the multitasking and multithreading capabilities of OS/2 made building a RDBMS easier. As a result, IBM announced a special version of OS/2, OS/2 Extended Edition, which included a communication subsystem and an RDBMS built using DB2 technology. The RDBMS component was called Database Manager. Applications running on a variety of IBM systems from inexpensive Intel-based PCs to minicomputers and large mainframes were now able to use the power of relational databases.

CLIENT/SERVER—THE BEST OF BOTH WORLDS

While IBM's RDBMS design provided the advanced facilities of relational database systems, such as structured query language (SQL), data integrity, data security, and transaction processing, these benefits were only available to the applications running on the same CPU as the RDBMS itself. With the growth of LANs and an increase



in popularity of graphical user interfaces fueled by inexpensive PCs, a new model for data access—client/server RDBMS—came into being. Under the client/server mode of operation, the relational database manager (database server) and the application that accesses data (database client) can reside on different machines connected via a telecommunication network.

In the client/server paradigm, client machines typically handle application-specific tasks and request data from database servers by passing an SQL request as a message to the database server. The results of each SQL command are returned over the network. The server processes the SQL request and returns only the requested data, rather than passing all the data back to the client to let it find what it needs, as is the case with file-based databases. The result is a more efficient utilization of distributed, processing power.

In 1990, OS/2 Extended Edition embraced the client/server paradigm by providing support for remote DOS and OS/2 clients. In 1992, the Database Manager component of Extended Services for OS/2 (successor to OS/2 EE) extended this support to applications written for Microsoft Windows 3.0. Also in 1992, IBM delivered Distributed Database Connection Services/2 (DDCS/2). DDCS/2 is an implementation of distributed relational database architecture (DRDA) that allows DOS, Windows, and OS/2 applications written for the Database Manager to access DB2, SQL/DS, and OS/400 data transparently. In effect, DDCS/2 brought the rest of the DB2 family into the client/server world.

WHAT'S IN THE NAME?

Because Database Manager for OS/2 had become a client/server RDBMS and is no longer a component of OS/2, it was transferred to IBM's Software Solutions Division, which is responsible for RDBMS direction. The IBM's Toronto Laboratory, responsible for SQL/DS, became the new home for Database Manager. In May 1993, a full 32-bit implementation of Database Manager with many of DB2's compatibility, performance, and quality enhancements was released. This new product is called IBM DB2 OS/2 (DB2/2), sold separately from the old Communication Manager component. As with OS/2 ES 1.0, DB2/2 is offered in a choice of two packages:

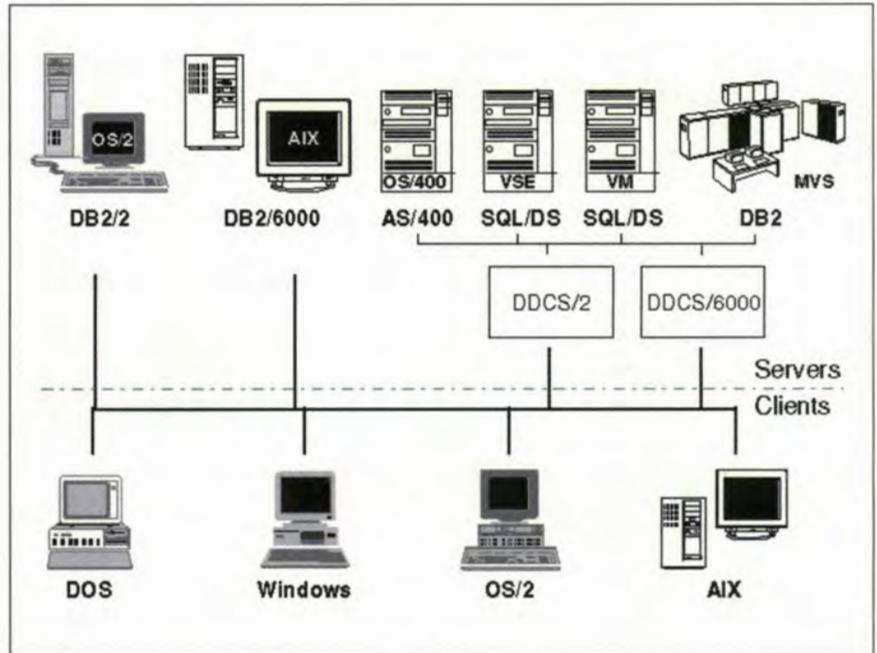


Figure 1. IBM DB2 family of client/server RDBMS

- **DB2/2 Single User.** Provides an RDBMS function for multiple, OS/2 applications sharing the PC with DB2/2.
- **DB2/2 Client/Server.** Provides a server-based RDBMS for multiple, DOS, Windows, and OS/2 applications running on remote, client PCs.

In July 1993, IBM extended its DB2 line of RDBMSs by introducing a UNIX-based database server, DB2/6000, for the IBM's RISC System/6000 family of processors running the AIX operating system. IBM's DB2 family is shown in Figure 1.

ACCESSING DB2 FAMILY DATA FROM OS/2

As a developer, you are interested in accessing the data stored in an IBM RDBMS from your OS/2 applications. IBM, together with a large number of third-party suppliers, offers tools that help you develop OS/2 applications that work with data stored in DB2 databases. IBM DB2 products provide two types of tools:

- **DB2 Family Development Tools And Utilities.** Tools and utilities for building applications that access DB2 data.
- **DB2 Family Application Enablers.** Run-time support for applications that are built using these tools.

DEVELOPMENT TOOLS

OS/2 applications that access DB2 data can be developed using one of two methods: embedded SQL or call-level interface (CLI) commands.

Embedded SQL will be familiar to programmers experienced with host database systems such as DB2 for MVS, DB2/VM, or VSE (formerly SQL/DS). SQL commands are entered as source-code statements. Because most compilers cannot process SQL statements directly, a precompiler converts source code into a form that can be processed by conventional compilers. While an extra precompilation step can be viewed as an inconvenience, embedded SQL is a preferred approach for many application developers because of its familiar, SQL interface. Embedded SQL can also generate static SQL for better performing applications.

CLI is based on the traditional, function-call method of invoking a function or a procedure to accomplish a task, in this case data access. Unlike embedded SQL, no preprocessing is required since compilers already support function calls.

IBM's implementation of CLI is based on an X/Open preliminary-draft specification. This allows users to migrate applications to other RDBMSs that support the same level of CLI. IBM's CLI is also compatible with the Open Database Connectivity (ODBC) specification for Microsoft Windows and allows the user to port Windows ODBC applications easily to OS/2 or other DB2 client platforms such as DOS and AIX.

IBM offers two products that provide tools and utilities for OS/2 application developers:

- DB2/2 (either single-user or client/server versions)
- DB2 Software Developer's Kit/2.

DB2/2. Both single-user and client/server versions of DB2/2 include an application-development environment for building 16- and 32-bit OS/2 database applications using FORTRAN and COBOL. DB2/2 provides three components that are of interest to application programmers:

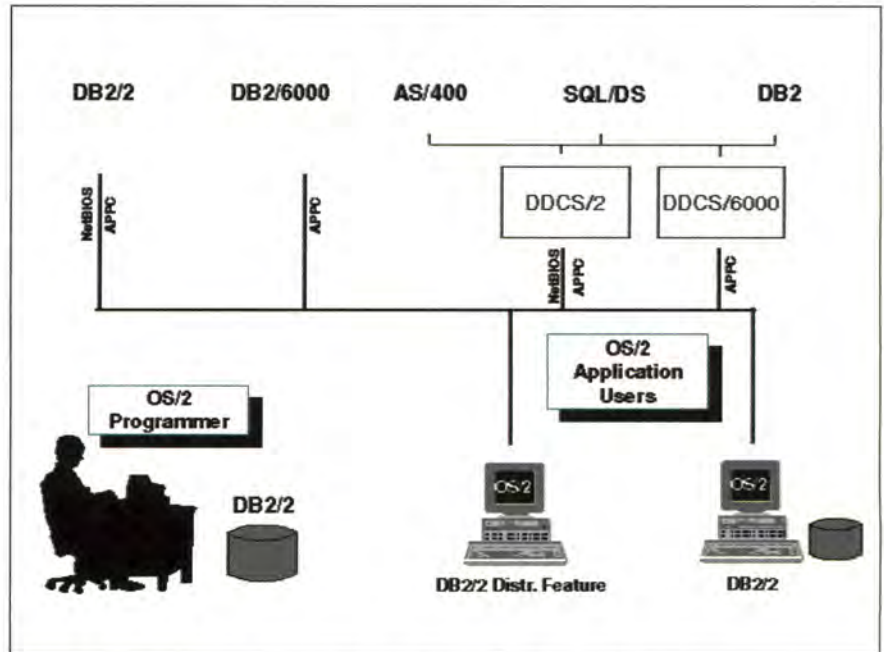


Figure 2. DB2/2 Client Application Enabler Distributed Feature environment

1. A relational database that can be used for testing applications without disrupting operations of a production server or changing operational data.
2. C, FORTRAN, and COBOL precompilers; code samples; and documentation for developing embedded SQL applications in those languages.
3. A command-line processor providing interactive, SQL capabilities that allows programmers to prototype SQL statements.

Applications developed using DB2/2 require that either DB2/2 itself or the DB2/2 Client Application Enabler Distributed feature be available in run-time environments.

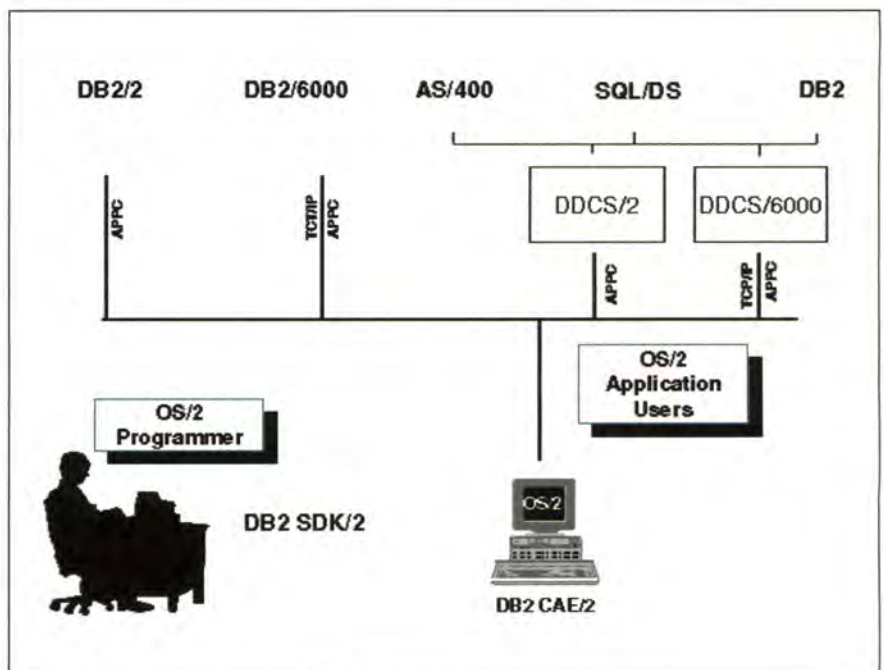


Figure 3. DB2 Client Application Enabler/2 environment

Fax at a higher level



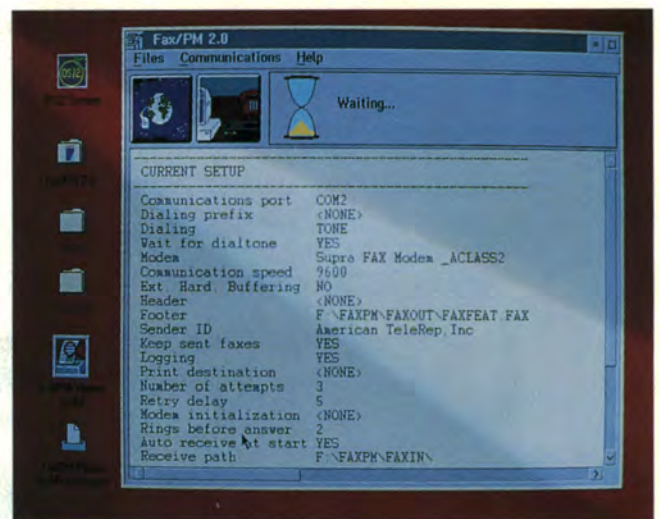
Whether you're in a Dos, Win-OS2 or OS/2 session running under OS/2 2.1, now you have the ability to send and receive faxes - without leaving the application. Just select the Fax/PM driver as your printer. To fax...just print!

Fax/PM 32-bit also really capitalizes on the advanced features of the Workplace Shell object-oriented interface. You can send faxes right from the Workplace Shell by simply dragging and dropping a data object to the fax object.

Features like these are too good to keep to yourself, so there are also Fax/PM versions available for LAN and Client-Server environments, all CID enabled.

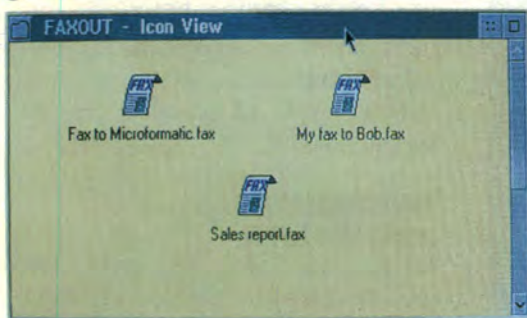
And applications developers will appreciate SOM APIs that allow for integration of the Fax/PM engine into custom applications. We could go on about how Fax/PM can make your PC an even more productive place.

But we'd rather let the fax speak for themselves.



with

Fax/PM.



To order or to find out more about Fax/PM, call U.S.: 1-203-644-1708 / fax 1-203-648-9587
Europe: (33) 1 48 70 19 00 / fax (33) 1 48 70 27 29



Microformatic

Europe
2 rue Navoiseau
F-93100 Montreuil
France.

U.S.A.
P.O. Box 722
610 Niederwerfer Road
South Windsor, CT 06074

Circle Reader Service Number 23



OS/2 and Workplace Shell are registered trademarks of IBM. Fax/PM is a registered trademark of Microformatic.



DB2 Software Developer's Kit/2. DB2 Software Developer's Kit/2 provides an environment for building 32-bit OS/2 database applications using C and FORTRAN. This product provides:

1. C and FORTRAN precompilers, code samples, and documentation for developing embedded SQL applications in those languages.
2. Programming libraries, code samples, and documentation for developing applications in C using the CLI API.
3. A command-line processor providing interactive, SQL capabilities so programmers can prototype SQL statements.

Applications developed with this product require that the DB2 Client Application Enabler/2 be present in run-time environments.

While the DB2 Software Developer's

Kit/2 does not provide tools and utilities for COBOL programmers, a comprehensive, COBOL development environment is available from third party suppliers such as Micro Focus Inc.

DB2 APPLICATION ENABLERS

The choice of DB2/2 or DB2 Software Developer's Kit/2 as a development environment is based on the application's run-time environment. Applications developed with DB2/2 require either DB2/2 or the DB2/2 Client Application Enabler Distributed Feature. Applications developed using the DB2 Software Developer's Kit/2 require DB2 Client Application Enabler/2 for run-time support.

The DB2/2 Client Application Enabler Distributed Feature provides users with run-time support for OS/2 applications developed using the tools and facilities of either DB2/2 or OS/2

Extended Services 1.0. It allows applications to access DB2/2 and, via DDCS/2, DB2 for MVS, DB2/VSE and VM, and OS/400. Both NetBIOS and APPC protocols are supported. Figure 2 shows a typical DB2/2 Client Application Enabler Distributed Feature environment.

DB2 Client Application Enabler/2 provides OS/2 users with run-time support for accessing the IBM RDBMS via TCP/IP and APPC protocols. To improve programmers' ability to deploy applications, DB2 Client Application Enabler/2 is offered under a "location license." This flexible, licensing scheme allows unlimited use on any number of OS/2 workstations of the DB2 Client Application Enabler/2 within a licensed customer location (a building identified by a unique street address). Figure 3 shows a typical DB2/2 Client Application Enabler/2 environment.

Take control of your OS/2 scheduling needs with

ATS

Version 2.0b

Advanced Task Scheduler for OS/2

Are you looking for a way to manage your production job streams? Do you have programs that need to run after the completion of one or more other programs? Do you want to process files that have just been received from another system or modified locally? Do you need to schedule around holidays and periods of heavy CPU load?

With ATS, you can accomplish all of this and MUCH MORE. With ATS you can run your programs when YOU want them to run and you don't even have to be there to start or monitor them.

Here are some of the features of ATS for OS/2:

- * Create True Job Streams
- * Programs can be scheduled to run anytime.
- * Programs can be dependent upon Files, Other Programs, Holiday Schedules, and External Signals
- * Complete Logging - To File and Console
- * Console Display of Running Tasks
- * Console Display of Holidays, Events and Tasks
- * Programming Interface for External Signals
- * Only \$249.00

MHR

Software and Consulting

2227 U.S. Highway #1 * Suite 146 * North Brunswick, NJ 08902

Voice: (908) 821 - 0359 Fax: (908) 821-0350

WHAT ABOUT REXX?

REXX has become a popular tool for many OS/2 developers, offering the simplicity of an interpreted batch language and the power of a serious programming language. DB2's family of products provides support for applications developed using REXX. Using new, visual-REXX development tools from WATCOM (VX-REXX) or HockWare (VisPro/REXX), REXX programmers can create sophisticated, graphical applications in a matter of days. Both visual-REXX products provide support for DB2.

PROGRAMMING AT A HIGHER LEVEL

In the past few years, we've seen growth of sophisticated, development environments, such as CASE and object-oriented tools. These promised reduced, application-development cycles and better, software quality. Many tools first appeared on OS/2 and included support for OS/2 Database Manager and now DB2/2. These included products like Easel, Enfin/3,

Circle Reader Service Number 21

Application Manager, and Choreographer. Many of them have been used with DB2/2 to build mission-critical applications for large corporations. IBM entered this market with products such as Personal Application System/2 and VisualAge/2. VisualAge/2 allows application programmers to build complex applications from a library of reusable parts. New programming tools are being developed or ported to OS/2 and enabled for DB2 access.

IN SUMMARY

The introduction of DB2/2 and DB2/6000 Client/Server RDBMS extended the IBM DB2 family to Intel and UNIX platforms. IBM and its business partners provide OS/2 programmers with a variety of tools for creating mission-critical applications that

work with this data. These tools cover a range of development environments from third-generation languages such as C, COBOL, FORTRAN, and REXX to object-oriented languages, such as SmallTalk, and visual-programming tools. DB2 technology is also available for applications that used PC databases in the past. For these applications, DB2/2 single-user offers the function and the same-strength database engine as the client/server version.

To help commercial-applications developers take advantage of DB2 RDBMS, IBM created the DB2 Developer's Assistance Program. Developers of commercial applications should call 1-800-627-8363 for more information.

Other articles in this DB2/2 series provide an introduction to developing OS/2 applications for DB2 RDBMS

using embedded SQL and an introduction to the EXPLAIN utility, which gives programmers and database administrators a better understanding of how DB2/2 processes queries and how to use this knowledge to improve performance of an application.

Leon Katsnelson, IBM Software Solutions Toronto Laboratory, Ont., Canada, is a product planner with DB2/2 and DB2/6000. He joined IBM Toronto Laboratory in 1985 as a software developer. Since then, he has held a number of positions in software development, testing, and product planning. He has been working in Database Technology Planning since March 1992, where he is responsible for client/server and DDCS strategy. He received a B.S. in computer science from York University, Toronto, Ont., Canada.



DON'T MISS OUT

on a single issue of OS/2

OS/2 Developer offers tips and techniques for more efficient computing for the advanced software developers who have recognized the need for 32-bit power.

TO ORDER

For new orders & customer service:

1-800-WANT-OS2

(1-800-926-8672) or

708-647-5960

FAX 708-647-0537

Looking for fast answers to your development questions?



Golden CommPass

OS/2 Navigation Software

You need to do two things. Go on-line with CompuServe Information Service. Use Golden CommPass to do it.

CompuServe hosts OS/2 developer and user forums, monitored by technical personnel. IBM developers are there with responses to anything that's got you stuck. Other OS/2 programmers and enthusiasts are there, too, comparing notes and giving feedback. It's definitely the place to be.

Time is money when you connect to CompuServe, and Golden CommPass saves you both. It lets you compose your questions off-line, send them in a flash and disconnect. It gets your replies while you're busy programming.

Golden CommPass keeps your development schedule on course. The fact is, the sooner you start using our program, the sooner they'll be talking about yours.



Creative Systems Programming Corporation

Post Office Box 961 • Mount Laurel, NJ 08054-0961

(609) 234-1500 • Fax: (609) 234-1920 • CompuServe: 71511,151



Circle Reader Service Number 22



Object-Oriented Programming

The SOMObjects Toolkit and DSOM offer opportunities to build new and powerful object-oriented applications. This article provides an introduction to DSOM programming. **BY CHARLES ERICKSON, NURCAN COSKUN, AND ROGER SESSIONS**

SOMObjects Developer Toolkit: Sharing SOM Objects with DSOM

One of the most powerful features of the second release of the SOMObjects Toolkit is the ability to allow access to SOM objects from multiple processes. The processes may be running within the same machine or on different machines connected together on a local, area network.

The extension to SOM that allows objects to be distributed among different processes is called Distributed SOM (DSOM). While DSOM may be considered an extension to the capabilities of SOM, it can also be called an Object Request Broker (ORB). DSOM fully complies with the Common Object Request Broker: Architecture and Specification 1.1 published by the Object Management Group (OMG) and x/Open. This specification is also known as the CORBA specification.

This article provides an introduction to DSOM programming by extending the appointment book example built in "SOMObjects Developer Toolkit: An Overview," (Nov./Dec. 1993). In that example, we implemented a client program that allowed us to add appointments to an appointment book. The appointment book and the appointments contained within it existed within one process. If we were to start a second client program, it would create its own, completely separate appointment book and appointments.

Suppose, however, that we want to share a single appointment book and its appointments among several, different client programs or among several instances of the same client program. Also, suppose we want the appointment book to be cre-

ated when the first client begins to use it. When other clients simultaneously use the appointment book, they should all use the one created by the first client. And finally, we'd like the shared appointment book to be destroyed only after the last, client program no longer requires access to the book. We envision some interprocess communications scheme, like shared memory, to solve this problem. However, we also want to allow client programs running on other machines access to the appointment book. We factor intermachine communications into the scheme. The whole idea sounds like a daunting project.

Instead of handling the interprocess communications ourselves, we use DSOM to share the appointment book. DSOM was created to solve many of these difficult problems, so that sharing objects between processes is not so difficult.

To share an appointment book using DSOM, we implement it using a DSOM server. A DSOM server is a program that, when running, houses the object that we want to share. A DSOM server is also a program that services requests by other processes to access the shared object.

DSOM provides a client/server model for programming applications. An object that is to be shared among processes is implemented by a server process. Client processes may make use of objects implemented at a server via remote method invocations. With DSOM, remote method invocations are made the same way as local method invocations. In many cases the only difference between using a local-object vs. a remote object is how the object is created and destroyed.



Charles Erickson



Nurcan Coskun



Roger Sessions



Before making the appointment book a shared object, we must first review what is involved in the implementation of a DSOM server. A DSOM server, in addition to being the process where the shared objects exist, is responsible for accepting client requests and converting them into method requests on objects within the server process. The server must return the output of method invocations to the processes that initiated them. A DSOM server also accepts requests to create and destroy objects at the server. These difficult jobs are taken care of by a few key objects provided in the DSOM framework. In the server process, these key objects are the SOMObject Adapter object (an instance of class SOMOA) and the Server object (an instance of class SOMDServer or a derivation of SOMDServer).

The SOMObject Adapter, better known as the SOMOA object, is created during the initialization of a server program. The SOMOA object handles communications with other processes. It accepts requests from client programs and converts them into method requests on objects in the server process. It also converts the output of method requests into a message that can be sent back to client programs. Only one instance of the SOMOA object exists in the server process.

Client programs use the Server object, among other things, to create and destroy objects in a server process. The Server object is an instance of the SOMDServer class or a class derived from SOMDServer. To create objects, such as the appointment-book object within a server, a client program invokes the `somdCreateObj` method on the Server object. To destroy an object, a client invokes the `somdDeleteObj` method on the Server object. Like the SOMOA object, the Server object is created during initialization of the server process, and only one instance of it exists.

These two objects do most of the dirty work in a DSOM server. To complete the implementation of a DSOM server, there are two final pieces to consider: a main program for initializing the server and processing incoming method requests and the implementation of a class library. The class library contains the implementation of the objects that will exist in the server process and will be shared among the clients of the server.

The DSOM server initialization and request processing that is done by the main program of a DSOM server can be handled the same way for many server implementations. To avoid the tedium

```
01 #include <somdserv.idl>
02
03 #include <apptbook.idl>
04
05 interface AppointmentBookServer : SOMDServer
06 {
07
08 #ifdef __SOMIDL__
09     implementation
10     {
11         // Class Modifiers
12         dllname = "apptbk.dll";
13
14         // Instance data
15         unsigned long numberOfUsers;
16         AppointmentBook apptBook;
17         // This is the one and only appointment book which can be used by
18         // clients of this AppointmentBookServer.
19
20         // Method Modifiers
21         somInit: override;
22
23         somdCreateObj : override;
24         // Creates an object of the specified class.
25         // If the object is of class "AppointmentBook", return the one
26         // and only AppointmentBook object.
27
28         somdDeleteObj : override;
29         // Dispose of the given object and any references to it.
30         // If the object is the one and only AppointmentBook managed by
31         // this server, free the object and set the apptbook attribute to
32         // NULL.
33     };
34 #endif /* __SOMIDL__ */
35 };
```

Figure 1. The IDL for the new Server object

of writing the same, main program time and time again, DSOM supplies a precompiled and linked server program named `somdsrv.exe`.

This server program sets up the DSOM run time, including the SOMOA object, and goes into a request loop. In the request loop, DSOM waits for a method request to arrive. When a request arrives, it is processed by the SOMOA object and dispatched by the Server object. If necessary, DSOM will dynamically load a dynamic link library (DLL) containing the implementation of an object. After the method



request is processed, DSOM returns to the top of the loop and waits for the next method request. The somdsrv.exe program satisfies the requirements of the shared appointment book, so we will use it in this example and say little more about writing a server program. For examples of server-program implementations other than the default somdsrv program, refer to the samples that are provided in the SOMObjects Toolkit.

The final piece of a server implementation is a class library for implementing the client-level classes. In this example, that would be the same DLL of appointment-book objects we built in our last article. At that time, it wasn't clearly apparent why we built our classes into a DLL; we could have linked our client-level class implementations directly into the client program just as easily. Now, because we want the DSOM server program to load the appointment book classes dynamically on demand, we need to package the classes into a DLL. DSOM can determine the DLL in which a class is implemented because of the following line in the IDL files:

```
dllname = "apptbk.dll";
```

In general, the SOM Frameworks all expect that class implementations can be dynamically loaded. Always build class libraries into a DLL.

Now, think about how the client program might make use of this DSOM server. In the earlier client program, a new appointment book was created with the following call:

```
AppointmentBook apptBook = AppointmentBookNew();
```

In the new client program, we want to create our object at the server. To do this, we can use the somdCreateObj method on the Server object. This method is a way for a client program to create an object at the server. Unfortunately, for every client that invokes somdCreateObj, a new AppointmentBook will be created. This is hardly what we had in mind when we built the server.

Ideally, a single appointment book is

```
01 SOM_Scope SOMObject SOMLINK somdCreateObj(AppointmentBookServer som-
Self,
02                                     Environment *ev,
03                                     Identifier objclass,
04                                     string hints)
05 {
06     SOMObject obj = NULL;
07     AppointmentBookServerData *somThis =
08                                     AppointmentBookServerGetData(som-
Self);
09     AppointmentBookServerMethodDebug("AppointmentBookServer",
10                                     "somdCreateObj");
11
12     if (strcmp(objclass, "AppointmentBook")==0) {
13 /* The object is an AppointmentBook, create a new one only
14    if we haven't already created one.
15    ----- */
16     if (_apptBook == NULL) {
17         _apptBook = parent_somdCreateObj(somSelf, ev, objclass,
hints);
18     }
19     _numberOfUsers++;
20     obj = _apptBook;
21 } else {
22 /* The object is not an AppointmentBook, create as always.
23    ----- */
24     obj = parent_somdCreateObj(somSelf, ev, objclass, hints);
25 }
26     return (obj);
27 }
28
29 SOM_Scope void SOMLINK somdDeleteObj(AppointmentBookServer somSelf,
30                                     Environment *ev, SOMObject
somobj)
31 {
32     SOMObject obj = NULL;
33     AppointmentBookServerData *somThis =
34                                     AppointmentBookServerGetData(somSelf);
35     AppointmentBookServerMethodDebug("AppointmentBookServer",
36                                     "somdDeleteObj");
37
38     if (somobj == _apptBook) {
39         _numberOfUsers--;
40         if (_numberOfUsers == 0) {
41             _apptBook = NULL;
42             parent_somdDeleteObj(somSelf, ev, somobj);
43         }
44     } else {
45         parent_somdDeleteObj(somSelf, ev, somobj);
46     }
47 }
```

Figure 2. The new Server object

Client/Server Database Solutions

It's available now – ready to perform on your desktop. A new function-rich, 32-bit relational database you can really trust with your growing client/server network, your mission – critical data *and* your business.

Introducing IBM DATABASE 2™ OS/2® (DB2/2™) from IBM Software Solutions, the birthplace of relational database technology.

DB2/2 includes an industrial-strength DB engine that supports transaction management, concurrency control, security, integrity, and recovery functions. Designed to exploit the power and open architecture of OS/2, it also supports industry-standard SQL for developing portable applications. And it runs your DOS, DOS Windows™ and OS/2 applications requiring *online* access.

You can access data directly from DB2/2 on your desktop or from a DB2/2 server on your LAN, and with

DB2



goes

DISTRIBUTED DATABASE
CONNECTION SERVICES/2,™
from DB2®, SQL/DS,™ and OS/400®
databases as if they were on your desktop, too.

This versatility can play a significant role in
an Information Warehouse™ solution
for your business.

We've developed an

desktop.



exciting demo diskette to show you just how well new DB2/2 performs – right on your desktop. Call us today for your free demo, or to order DB2/2: 1-800-342-6672; or fax: 1-800-445-2426. In Canada, call 1-800-465-7999, ext. 850. An upgrade from OS/2 Extended Edition or Extended Services is also available.

IBM, OS/2, DB2 and OS/400 are registered trademarks and DATABASE 2, DB2/2, DISTRIBUTED DATABASE CONNECTION SERVICES/2, SQL/DS and Information Warehouse are trademarks of International Business Machines Corporation. Windows is a trademark of Microsoft Corporation. © 1993 IBM Corp.


```

DLLOBS = appt.obj mtg.obj ccall.obj apptbook.obj booksvr.obj \
        initmod.obj

booksvr.ih: booksvr.idl
booksvr.obj: booksvr.ih booksvr.c

initmod.obj: initmod.c appt.h mtg.h ccall.h apptbook.h booksvr.h

sample.obj: sample.c appt.h mtg.h ccall.h apptbook.h booksvr.h
$(CC) $(CFLAGS) -c $.c

```

Figure 3. Changed and added lines of the make file

```

01 impl:
02     regimpl -A -i apptBookServer -v AppointmentBookServer
03     regimpl -a -i apptBookServer -c Appointment
04     regimpl -a -i apptBookServer -c Meeting
05     regimpl -a -i apptBookServer -c ConferenceCall
06     regimpl -a -i apptBookServer -c AppointmentBook

```

Figure 4. The regimpl utility

shared among all client programs, regardless of which client started first. The server would free the object when the last client program no longer needs access to the appointment book. To do this, we'll participate in the creation and destruction of the appointment-book object. DSOM was designed with this in mind. To participate in the creation and destruction of objects at the server, we can create our own `Server` object derived from the DSOM supplied `SOMDServer` class. Because the class name of the `Server` object is determined at run time, we can still use the generic, server program, `somdsrv.exe`.

In our new implementation of a `Server` object, which we'll call an `AppointmentBookServer`, we want to intervene when a client attempts to create an `AppointmentBook`. To do this, we override the `somdCreateObj` method of `SOMDServer`. We also want to be involved when a client attempts to destroy an

We'd like to say a word about client/server development.

TESTING.

You've chosen the platform, architecture, development tools, and application. Now's the time to decide how you're going to test your client/server applications.

The Softbridge Automated Test Facility (ATF) lets you fully automate regression testing, letting you test "often and early" in the development cycle. And that means lower costs, decreased time to release, and higher quality results.

If you're building client/server (or stand-alone) applications under OS/2, Windows, or NT, you should be using ATF to test them. To learn more about ATF, call 800-955-9190.

 Softbridge, Inc.
617-576-2257 (Phone)
617-864-7747 (FAX)

ATF: The final word in testing.

The easy to use 32 bit OS/2 PM IPF Language Editor

IPF Editor

- Add help to applications in minutes, not days
- Designed for easy use by programmers and non-programmers
- Generate C and Resource source files to add help to applications automatically
- Import wordprocessing files, including WordPerfect, quickly and accurately
- Create IPF tables from 1-2-3 and Excel spreadsheets
- Preview IPF file output without compiling
- Create all hypertext links automatically
- Support for HFPS long file names
- Create multimedia help and on-line documents with optional voice and animation modules
- Compatible with both Netware and LAN Server
- Includes complete on-line help and documentation

Introductory Price: \$95
With Printed Manual: \$125

Perez Computing Services
4725 Monte Vista Place
Mount Vernon, WA 98273

Orders and Information
(206)428-5025
on compuserve at
70410,2416

IPF Editor and PCS are trademarks of Perez Computing Services.
All company and product names are trademarks of their respective owners.

Circle Reader Service Number 26

AppointmentBook. To do this, we override the `somdDeleteObj` method. The IDL for the new `Server` object is contained in Figure 1.

By now, you should be familiar with deriving a new class and overriding methods. In line 1 of Figure 1, we include the IDL definition of the `SOMDServer` class. In line 5, we name the new class and indicate that it is derived from `SOMDServer`. In line 12, we specify that we plan to put the implementation of this new class in the same DLL as all our other classes. In lines 15 and 16, we declare some internal, instance data for the `Server` object and, in line 21, we override `somInit` so that we can initialize the instance data when the `Server` object is instantiated. Finally, in lines 23 and 28, we override the `somdCreateObj` and `somdDeleteObj` methods so that we can participate in the creation and destruction of objects in the server.

We've purposely chosen to keep our `Server` object very simple. Our `Server` object, therefore, will only support one `AppointmentBook` object. The `apptBook` declaration in line 15 will be the one and only `AppointmentBook` that will exist at the server. It is possible, even desirable, to support more than a single `AppointmentBook` object in the server, however this would require that we maintain a table of `AppointmentBook` objects. For convenience, we've chosen the simpler implementation.

The `numberOfUsers` instance data in line 16 of Figure 1 is used to keep track of how many client programs are currently using our one `AppointmentBook` object. We need to keep track of this information so we know when it is possible to free the `AppointmentBook`. When the counter is decremented to zero in the `somdDeleteObj` method, we'll free the `AppointmentBook`. While the counter remains one or more, we'll neither create nor free our single

appointment book, but simply increment or decrement the number of users.

The implementation of our new `Server` object is shown in Figure 2.

To save space, we do not show the implementation of the `somInit` method. `somInit` simply initializes the `apptBook` to `NULL` and sets the `numberOfUsers` to zero.

In line 12 of the `somdCreateObj` method, a check is made to see if the object about to be created has a class name equal to `AppointmentBook`. If it does and the `apptBook` instance data is `NULL`, a new `AppointmentBook` object is created in line 17 by invoking the parent implementation of the `somdCreateObj` method. Whether a new `AppointmentBook` object is created or the existing `AppointmentBook` object is used, the number of users of the book is incremented in line 19.

The `somdCreateObj` method will also be used to create the other objects in our server process. This includes the `Meeting`



OS2TREE™

See your entire system graphically at a glance!
"What XTREE GOLD is to DOS and More!
And What Norton Commander Missed"

- Completely programmable: every function/key can be customized!
- Display your tree structure for all system and LAN disks concurrently!
- Manipulate your files, directories, entire disks and/or entire system!
- HPFS fully supported
- Pop-up list of files for any directory
- Edit/browse file(s) using any EDITOR or PROGRAM!
- Extensive FILE Search and Tree directory search capabilities make locating files and/or data easy.
- Upload/download files between mainframe and PC
- Secure delete (sensitive files wiped clean)

Intro Price: \$79.99

Professional Version: \$249.99

Limited time special \$150.00

Network Licenses Available

LEVINE COMPUTER CONSULTING SERVICES

7640 Provincial Dr., Suite 213, McLean, VA 22102

Orders only (800) 383-9495
FAX/Inquiries/HelpLine (703) 790-1660

Circle Reader Service Number 27

Includes
32-Bit
OS/2 2.x
DLL

\$159

dbfLIB

Programmer's Library

Accessing dBASE files has never been easier!

Features:

- ↳ Royalty free DLLs for OS/2 and Windows
- ↳ File and record locking APIs
- ↳ Handle based calls
- ↳ Memo field & index support
- ↳ Static libraries for DOS, Windows, and OS/2

Coming soon...
February 1994
- More index formats
- New APIs
- REXX support
- Windows NT DLL
- Much more...

dbfLIB gives C programmers the easy task of accessing dBASE files while maintaining the speed and size of C programs. The APIs in dbfLIB are fully compatible across DOS, Windows, and OS/2.

**Visa and MasterCard
accepted. (713) 537-0318**



dSOFT Development Inc.
4710 Innsbruck Drive
Houston, TX 77066

Circle Reader Service Number 28

and ConferenceCall objects contained in the AppointmentBook. When the object to be created is one of these other objects, it is created the normal way, in line 24, by simply invoking the parent implementation of the somdCreateObj method. We don't need to handle these types of objects in a special way, like the AppointmentBook object; because they are contained within the one AppointmentBook

object.

When an object is to be destroyed with somdDeleteObj, we first check to see if it is our AppointmentBook object. This check is made in line 38 by comparing the given object to the value of apptBook. We only want to free the AppointmentBook if there are no other current users of the book. This check is made in line 40. When there are no other users, the apptBook is nulled so

that the next invocation of somdCreateObj for an AppointmentBook will create a new one. Then, the AppointmentBook object is freed in line 42 by invoking the parent implementation of somdDeleteObj.

When the object to be destroyed is anything other than our AppointmentBook object, it is destroyed in the normal way in line 45 by invoking the parent implementation of somdDeleteObj.

COMPLETING THE IMPLEMENTATION

We are now prepared to complete the implementation of our DSOM server. All that remains is for us to package the new AppointmentBookServer class into our dynamically loadable class library (APPTBK.DLL) and "install" our DSOM server implementation.

To add the new class to the APPTBK.DLL, we need to make a couple minor changes to the make file shown in our previous article, add a line to the SOMInitModule procedure, and add the class externals to the APPTBK.DEF file. The changed and added lines of the make file are shown in Figure 3. Note the additions to build and use booksvr.obj, booksvr.h, and booksvr.i.h.

The only change required to SOMInitModule is the addition of the following line to the body of the function:

```
AppointmentBookServerNewClass(0,0);
```

and the addition of the class header file at the beginning of the C file:

```
#include <booksvr.h>
```

The additions to the EXPORTS portion of the APPTBK.DEF file are:

```
AppointmentBookServerCClassData
AppointmentBookServerClassData
AppointmentBookServerNewClass
```

The procedure for installing a DSOM server involves more than simply copying the server program and class library into the PATH and LIBPATH. We must also:

1. Set some environment variables used by DSOM at run time.
2. Register the classes we plan to access from other processes in a database

SINGLE KEY-STROKE SCREEN PRINTING



CLIPBOARD
VIEWER

SCREEN
CAPTURE

- Quality Color or Black & White copies of color Desktop Images.
- Copy any portion of any image on the Desktop.
- Screen Saver prevents monitor burn-in.
- Date & Time Display for Time-Stamping images.
- LAN installable. Complete support for LAN attached printers.
- Economical combination of important utilities. Entity licensing available.
- The most stable and reliable product available.
- Includes both OS/2 2.1/2.0 (32-bit) and OS/2 1.3 (16-bit) versions.

THE LEADER: PrntScrn™ Screen Image Utility for OS/2



MITNOR Software
Phone: (918) 357-1628
Fax: (918) 357-2869



- known as the interface repository.
3. Register a description of the server implementation in a database known as the implementation repository.

The following is a description of the environment variables that DSOM relies upon:

- **SOMIR** is a list of files, delimited by semi-colons, that together make up the interface repository. This variable is set when you install DSOM.
- **SOMDDIR** is the name of the directory where the implementation repository is stored.
- **USER** identifies the user ID of a DSOM client application.
- **HOSTNAME** identifies the name of the machine on which DSOM is running. If the client and server are running on the same machine this variable may be set to any value. We typically set it to `localhost` in this situation.
- **SOMSOCKETS** identifies the name of the class used to implement sockets for DSOM. If using TCP/IP, set this to `TCPISockets`. This variable is only required if you are using the `SOMObjects Workgroup Enabler`.

For DSOM to handle the remote method invocations that will be made on our library of appointment book classes, it must know every detail about the methods in the class library. It must know the names of the methods of the classes, the return types of the methods, the number and types of parameters the methods take, and so on. This information is stored in the interface repository. The interface repository is built and maintained by the SOM compiler when we include the `-u` switch as we invoke the compiler.

The implementation repository contains information about a DSOM server implementation. This includes the name of the program that implements the server, a user-friendly alias that can be used to refer to the server, the class name of DSOM Server object, and the names of the classes implemented by the server. DSOM uses the implementation repository to find and, if necessary, start a server program.

DSOM provides a utility program, named `regimpl.exe`, to build the imple-

mentation repository. Since we are using the DSOM-supplied, generic server program, `somdsrv.exe`, for our appointment-book server program, we just need to specify the name of our server, the class name of our Server object, and the classes that it implements. Our use of the `regimpl` utility is shown in Figure 4. Line 2 adds a new implementation named `apptBookServer` to the implementation repository.

Client programs will refer to the DSOM server with this alias name. In the same line, we specify that the class name of the Server object is `AppointmentBookServer`. By specifying the name of the Server object, DSOM knows to use our class rather than the default `SOMDServer` class. Lines 3 through 6 specify the classes our server supports.

After one final step that must be com-

USE IT

The **GammaTech Utilities** are designed to enhance and complement your OS/2 system. These utilities provide the ability to perform vital maintenance and recovery operations easily without extensive technical knowledge. Use **GammaTech Utilities** for your critical data or you might lose it.

GammaTech Utilities for OS/2

or

LOSE IT

- Undelete Erased Files
- Optimize HPFS and FAT Volumes
- Backup INI and Desktop Files
- Recover HPFS Volumes
- Protect and Backup Boot Sectors
- Identify and Mark Bad Sectors
- Reconstruct Boot Sectors
- Protect/Lock Files
- Permanently Erase Sensitive Data
- Mass Delete Selected Files
- Sort FAT Directories
- View and Edit Selected Files
- Disk Sector Edit (ASCII or Hex)
- Display Volume Information
- Alter File Attributes
- Add Comments to Files
- Display File Fragmentation
- Display Directory Information
- SAA/CUA Compliant PM Interface

For OS/2 Version 2

VISA, MasterCard, American Express, C.O.D. • Overnight Service Available

(405) 947-8080 • Fax (405) 632-6537

SofTouch Systems, Inc., Workstation Division

1300 S Meridian, Suite 600, Oklahoma City, Oklahoma 73108





pleted on the server machine, our DSOM server will be ready for use. A process called the DSOM daemon must be started. The DSOM daemon is used to connect DSOM client processes to a DSOM server process and start DSOM servers automatically, if necessary. At the command prompt or in the STARTUP.CMD, enter the command:

```
start somdd
```

When a client program attempts to use our server, named `apptBookServer`, the DSOM daemon will consult the implementation repository to determine the name of the server program to run. Because we didn't specify one, it will start the default server program, `somdsrv.exe`.

All that remains to complete the example are the changes to our earlier client program, which are required to use the DSOM server. In the previous client program, we created the appointment book with the following line:

```
apptBook = AppointmentBookNew();
```

To create the appointment book in the server, we replace this line with the following lines:

```
SOMD_Init(ev);
server = _smdFindServerByName
    (SOMD_ObjectMgr, ev, "apptBookServer");
apptBook = _smdCreateObj(server, ev,
    "AppointmentBook", NULL);
```

The first line initializes the DSOM run-time environment by calling `SOMD_Init`. During initialization, an object called the DSOM Object Manager is created. The Object Manager object is stored in the global variable `SOMD_ObjectMgr`. The Object Manager provides a variety of useful functions for client programs including methods for finding Server objects.

In the next line, we use the Object Manager object to find our server, named `apptBookServer`. It is important to understand that the Server object returned by the `smdFindServerByName` method is not actually the same object that exists in the server process. What is returned to the

client is an object that has the same interface as the actual object in the server but an implementation provided by DSOM. DSOM calls these special objects "proxy objects." When we invoke a method on a proxy object, the DSOM implementation for that object packages-up the parameters we have passed to the method into a message. The message is then forwarded to the server process where the real object exists. The server's main program wakes-up when it receives this message and passes it to the server's SOMOA object for resolution into a real-method call on the real object.

DSOM builds a client proxy object every time it returns an object from a server. Therefore, just as the server object returned from the call to `smdFindServerByName` is a proxy object, so is the `apptBook` object returned by the call to `smdCreateObj` in the next line of our program. Keep in mind that the `smdCreateObj` method has actually been executed in the server process. The output of the method is the `AppointmentBook` object. When the output of a method invoked on a proxy object is an object, DSOM will convert it into a client proxy object automatically and transparently.

The next change to the client program is how we create the appointment objects, which will be added to the appointment book. In the standalone client, we used the following to create a new Meeting object:

```
appt = MeetingNew();
```

In the DSOM client, we create the Meeting object just as we did the AppointmentBook object, with `smdCreateObj`:

```
appt = _smdCreateObj(server, ev,
    "Meeting", NULL);
```

ConferenceCall objects are created in the same way.

A proxy object can be used just like a normal object and, in fact, can be used as a parameter in method invocations on other proxy objects. For example, in the following line, we use the `appt` proxy as an argument in the `addAppointment` method invoked on the `apptBook` proxy

object:

```
id = _addAppointment(apptBook, ev,
    appt);
```

When a proxy object is passed to a remote object, the server locates the real object and uses it instead of the proxy when the method is invoked.

Most interactions with remote objects are the same as with local objects. However, client programs must perform some memory management when they invoke methods on remote objects that return allocated space such as strings, sequences, arrays, and objects. In the following line, the `bufferize` method is invoked to return a formatted string, which can be displayed to show the contents of an Appointment object:

```
buffer = _bufferize(sequenceElement
    (apptList, i), ev);
```

Recall that the `bufferize` method returns a string that has been allocated within the Appointment object. Because the pointer is only valid within the address space of the server process, it can't be returned to the client program. In this case, DSOM allocates equivalent space within the client process.

This storage must be managed by the client program. When the buffer returned by `bufferize` is no longer needed, it must be freed by calling the `ORBfree` function:

```
ORBfree(buffer);
```

A similar, memory-management issue occurs during the invocation of the `getDaysAppointments` method:

```
apptList = _getDaysAppointments(appt-
    Book, ev, year, month, day);
```

This method returns the sequence `apptList` that contains Appointment objects for the given day. Recall that a sequence is a structure that contains a buffer field, which is a pointer to an array. Like the `bufferize` method, this method returns a pointer to an array allocated within the object. Therefore, we must free the

A Blue-Ribbon Panel Named **POWER TRANSLATOR™ PROFESSIONAL** Most Innovative Software of 1993...



But our users are saying things that *really* make us proud.

The prize: 1993 Discover Award
for most innovative software.

The judges: John Dvorak,
Michael Miller, Marvin
Minsky, Esther Dyson, and
Stewart Cheifet. Their Affilia-
tions: PC Magazine, PC Computing,
MacUser, Computer World, Computer Chronicles

and MIT. The winner: **POWER TRANSLATOR
PROFESSIONAL** foreign language translation
software, from Globalink®!

But our users don't think about "innovation."
They have specific work in mind.

For Eric Vogt, translator for the American
Red Cross, **POWER TRANSLATOR PROFESSIONAL**
guarantees consistency and "...saves an enormous amount
of time... A person simply can't remember 5,000 scientific
terms and how they translated each of them the last time."
For Jamala Banks, a language student at Howard Univer-
sity, "what you would normally take a day on, you can

"...it saves an enormous amount of time..."

— Eric Vogt, American Red Cross translator

"...I really expanded my vocabulary, my grammar
...using the program has been just great."

— Jamala Banks, Howard University student

now virtually do in just minutes!"

Available for DOS, OS/2®,
Macintosh® and UNIX®, **POWER
TRANSLATOR PROFESSIONAL**
features speeds up to 100 times
faster than human translators,
easily-updated dictionaries of over
250,000 words and phrases and a 90%-plus
accuracy rate! Subject dictionaries are also available, in-
cluding computer, legal, business and finance.

Find out about the software package that's got Discover
Magazine – and a lot of other people – talking. Call
Globalink today and ask about **POWER TRANSLATOR
PROFESSIONAL**, your ticket to world-wide communication.



for OS/2



lobalink, Inc.

9302 Lee Highway, Fairfax, VA 22031
Intl.: 1-703-273-5600
Fax: 1-703-273-3866

1-800-767-0035

Incredibly accurate translations: English to/from Spanish, French, or German



REFERENCES

SOMobjects Developer Toolkit Users Guide, (IBM Doc. 96F-8649), 1993.

The Common Object Request Broker: Architecture and Specification, Revision 1.1, OMG Document Number 91.12.1, Object Management Group and x/Open, Framingham, MA, 1992.

Rhetorex Voice Processing boards make CTI a reality.



If you're asking "what's CTI," you're missing one of the hottest new technologies going.

Computer Telephony Integration links PC-based computer applications to the telephone network, providing voice/fax mail, interactive voice response, and other telephony applications.

Interested? Maybe you're already developing a CTI application. Then it's time to discover Rhetorex.™

The fact is, when it comes to multi-port voice processing components, no one offers more value than Rhetorex.

With Rhetorex voice platforms, you'll get state-of-the-art DSP-based technology, designed from

the ground up for reliability. Sophisticated DSP algorithms give you unparalleled performance. Best of all, enhancements to algorithms are made via software upgrade.

And Rhetorex products help you develop applications quickly. All components include device drivers, a straightforward API, and sample programs. Friendly, free technical support is available at any stage of development.

For the best value in CTI technology—from our 2 and 4 port voice processing boards and fax boards, to our 24-port platform—then give Rhetorex a call. (408) 370-0881. And start making CTI a reality.



RHETOREX

Rhetorex, Inc.
200 E. Hacienda Avenue
Campbell, CA 95008-6617
Tel. (408) 370-0881
Fax (408) 370-1171

All trademarks identified by the ™ symbol are trademarks of Rhetorex, Inc.
All other trademarks belong to their respective owners. © 1993 Rhetorex, Inc.

```
array:  
ORBfree(apptList._buffer);
```

Before we can free the array, however, we need to free the objects contained within the array. This is because the objects returned in the sequence are proxy objects. These objects were created automatically when DSOM returned the results of the method invocation. These objects can be freed with the following code:

```
for (i=0; i < sequenceLength(apptList); i++)  
_smdProxyFree(sequenceElement(apptList,i), ev);
```

If a DSOM client program ignores these memory-management responsibilities, eventually it will run out of free memory.

To terminate the client program, we replace the `smdFree` method call with these three lines:

```
_smdDeleteObj(server, ev, apptBook);  
_smdProxyFree(apptBook, ev);  
SOMD_Uninit(ev);
```

In the first line, the `AppointmentBookServer` object is instructed to destroy its `AppointmentBook` object. Recall that our overridden implementation of the `smdDeleteObj` method will only free the `apptBook` if there are currently no other clients using the `AppointmentBook`. The method call also reduces the number of users of the `AppointmentBook` by one.

`smdDeleteObj` takes care of the `apptBook` object at the server, however we still need to free the local, proxy object. This is accomplished with the call to `smdProxyFree`.

Finally, before terminating the program, a DSOM client program must return any resources used by DSOM to the operating system. This is done by uninitializing the DSOM run time with a call to `SOMD_Uninit`. DSOM client programs, if possible, should include a call to `SOMD_Uninit` within an `existList` handler.

While this example application is quite simple, the problems that it

illustrates and solves are very realistic. The SOMobjects Toolkit and DSOM offer opportunities to build new and powerful, object-oriented applications and is available now for both OS/2 and AIX. In the future, watch for SOM and DSOM to be available for Windows applications.

Nurcan Coskun, IBM Corp., Austin, Texas, has expertise in integrated-programming environments, code generators, incremental compilers, interpreters, language-based editors, symbolic debuggers, application frameworks, and language design. She has worked on the OS/2 Database Manager and is now working on object-oriented programming environments. She holds a B.S. in industrial engineering from Middle East Technical University and an M.S. and Ph.D. in computer science from the University of Missouri, Rolla. Coskun can be contacted on Internet at nurcan@ausvm1.vnet.ibm.com.

Roger Sessions, IBM Corp., Austin, Texas, is the author of Reusable Data Structures for C. His most recent book, Class Construction in C and C++: Object-Oriented Programming Fundamentals, was chosen as a main selection for the Library of Computer and Information Science book club. He has authored many articles and frequently lectures on object-oriented programming, C++, and SOM. He is working on persistent-SOM frameworks and has previously worked with high-performance relational databases and object-oriented storage systems. Sessions has a B.A. from Bard College and an M.E.S. in database systems from the University of Pennsylvania. He can be contacted on Internet at roger@vnet.ibm.com.

Charles Erickson, IBM Corp., Austin, Texas, is an advisory programmer who is currently developing SOM frameworks in IBM's Object Technology Products group in Austin. He joined IBM in 1984 and helped develop the 5250 emulators in PC Support and the OS/2 Communications Manager. He received a B.S. in computer science from Iowa State University.

DON'T MISS OUT



on a single issue of OS/2

OS/2 Developer offers tips and techniques for more efficient computing for the advanced software developers who have recognized the need for 32-bit power.

TO ORDER

For new orders & customer service:

1-800-WANT-OS2

(1-800-926-8672) OR

708-647-5960

FAX (415) 905-2233

Written subscription orders:

OS/2 Developer • P O Box 1079

Skokie, IL 60076-8079

Subscription rates for 6 issues:

☐ U.S. — \$39.95

☐ Canada — \$55.95

International surface mail

☐ International air mail — \$69.95

☐ Check inclosed

Charge my:

☐ Visa ☐ MasterCard ☐ AmEx

Card #

Exp. Date

Signature

Name

Company

Address

City

State/Zip

Make checks payable to Miller Freeman, Inc. Foreign orders must be in U.S. funds. Please allow eight weeks for subscriptions to begin.

HOUSAD



Object-Oriented Programming

*The user interface class library of the IBM C Set++ compiler revises and simplifies the programming interface to the popular notebook control. Instead of mastering various Presentation Manager messages and notifications, developers creating 32-bit OS/2 applications with the new interface use simple objects to organize, manipulate, and present information. This article does not attempt to explain every method of the new interface but instead gives an in-depth tour of the most-used types of functions. By **BRAD BROYLES***

Building a Notebook with IBM C Set ++ Objects

The notebook control introduced in IBM's OS/2 2.0 (and in the IBM SAA Common User Access Controls Library for OS/2 1.3 and Windows 3.x) is a very popular and easy-to-use way of organizing data into a format that is simple for users to understand and manipulate. In addition to providing pages that allow developers to organize and separate information into different categories, the notebook control supplies page tabs that users can manipulate to move among the different categories with either a pointing device or the keyboard. By supplying objects that represent the notebook control and can be used as keys for inserting and modifying notebook pages, the user interface class library of the C Set ++ compiler simplifies the coding interface and allows developers to approach the notebook control from an object-oriented point of view.

CREATING A NOTEBOOK OBJECT

The *INotebook* Class. The *INotebook* class is the heart of the notebook control. An instance of the *INotebook* class must be created before any pages can be inserted into the notebook. The *INotebook* class has three constructors, any of which can be used for creating notebook objects. The first constructor provided by the class supplies a default rectangle and notebook style so that only a control ID, parent window, and owner window have to be specified, as shown in Figure 1.

ID_NOTEBOOK is the control ID of the created notebook; *pParent* is a pointer to the *IWindow*

object, which will be the parent of the notebook window; and *pOwner* is another *IWindow* pointer that points to the window, which will be the owner of the notebook (when these two pointers are the same, there is another *INotebook* constructor that takes one pointer and assumes it points to the parent and the owner).

Since neither a rectangle nor a notebook style was specified, this constructor will provide a default sizing rectangle and the default notebook style (a visible notebook with solid binding on the left, back pages drawn on the bottom right, square page tabs with major tabs on the right, left-justified status text, and centered tab text). If it was necessary to specify a particular rectangle for the notebook's initial size or a special notebook style, those parameters could be added to the constructor call. The size and style of the *INotebook* instance can be changed after creation with the *setStyle* and *setSize* methods, which *INotebook* inherits from *IWindow*, one of its superclasses. However, a more direct way of altering the notebook's appearance is to use the style functions that the *INotebook* class provides. For example, the *setBinding* method provides an interface for changing the notebook's binding style, and you can use the *INotebook*'s *binding* method to query the current binding style of the notebook (*INotebook* enumerations are used to define the different types of bindings). The code in Figure 2 uses the *INotebook* style methods to change the notebook object created previously from the default style to a spiral-bound notebook with polygon page



tabs, major tabs on the bottom, and back pages drawn to the bottom right.

PAGE MANIPULATION

Inserting Notebook Pages. Typically, each page of a notebook displays a window or dialogue object, which in turn contains other control objects such as text, entry fields, or push buttons. The notebook object contains no pages when it is created (although the drawn back pages make it appear this way). The `PageSettings` object is used to specify the features of a page when the page is added to the notebook. For example, the `PageSettings` object indicates whether or not the page should have a major tab, minor tab, or status line on it; what tab text, status text, or user data should be associated with the page; and if the page is auto-sizing (if so, the window or dialogue displayed on the page will be automatically sized to fit; if not, the application will be responsible for handling the size). The values in a `PageSettings` object are used only when the page is added to the notebook; changing a `PageSettings` object after a page has been added has no effect on the page.

Adding a page is a function of the notebook, and the `INotebook` class provides several methods for adding pages. Each one takes a `PageSettings` object that defines the attributes of the page to be added and a pointer to the window or dialogue object, which should be displayed on the page after it is added to the notebook (if this pointer is null, then an empty page will be added; empty pages can still have status lines and a tab but do not have dialogues displayed on them). Some of the methods take an additional parameter for adding pages before or after another existing page.

The code in Figure 3 illustrates how a `PageSettings` object can be used. One `PageSettings` object will be used to add three pages to the notebook that was previously created. In this example, there are three dialogues loaded from the application's user resource library and added to the notebook (these dialogues have successive IDs).

The `PageSettings` object is created with the combined attributes of `autoPageSize` and `majorTab`; all the pages added with this `PageSettings` object will handle sizing themselves and have major

```
/* Create a new INotebook object and save a pointer to it */  
  
pNotebook = new INotebook(ID_NOTEBOOK, pParent, pOwner);
```

Figure 1. Using an `INotebook` constructor

```
/* set the binding, tab shape, and orientation of the notebook */  
  
pNotebook->setBinding (INotebook::spiral);  
pNotebook->setTabShape (INotebook::polygon);  
pNotebook->setOrientation (INotebook::backpagesRightTabsBottom);
```

Figure 2. Altering an `INotebook` object's appearance

```
long i = 0;  
/* create an IPageSettings object with the page-sizing and major  
tab attributes */  
INotebook::PageSettings pageSettings  
    (INotebook::PageSettings::autoPageSize |  
     INotebook::PageSettings::majorTab);  
for (; i < 3; i++)  
{  
    pDialogs [i] = new IFrameWindow (ID_FIRSTPAGE + i); //load the  
    dialog  
    pageSettings.setTabText (IString (i + 1));           //set tab  
    text  
    pNotebook->addLastPage (pageSettings, pDialogs [i]); //add to  
    notebook  
}
```

Figure 3. Inserting notebook pages

tabs. Inside the `for` loop, after the dialogue is loaded, the `setTabText` method is used to define tab text for the `PageSettings` object. This tab text will be passed to the notebook when the page is added. After the tab text is set, the notebook's `addLastPage` method is called, a new page is added to the notebook, the attributes and the tab text that were previously set in the `PageSettings` object are transferred to the new page, and the dialogue currently referenced in the `pDialogs` array is displayed on the page. Once the page has been added, the `PageSettings` object can be changed without affecting the new notebook



page—there is no connection maintained between the two. This allows the tab text in the `PageSettings` object to be changed the next time through the loop without affecting the page that was added in the previous iteration. After the three pages have been added, the notebook looks like Figure 4.

Altering Notebook Pages. Since there is no linkage between the `PageSettings` object and a page after it is added to the notebook, the `INotebook` class provides methods for altering pages. To identify which page in the notebook is to be changed, use the `IPageHandle` object. In the same way that handles in Presentation Manager are used to identify objects like windows, `IPageHandle` objects are used to identify individual pages in a notebook object.

The `INotebook` class returns `IPageHandle` objects from various query methods such as `firstPage` (returns an `IPageHandle` referencing the first page of the notebook), `lastPage` (returns an `IPageHandle` referencing the last page of the notebook), and `topPage` (returns an `IPageHandle` referencing the page currently visible). Similarly, if an instance of `IPageHandle` is already available, the `nextPage` and `previousPage` methods can be used for obtaining an `IPageHandle` for the page following or preceding the page referenced by the existing `IPageHandle`.

In the code sample in Figure 5, an `IPageHandle` object referencing the last page of the notebook is returned by the notebook and used to change the tab text of that page.

Traversing Notebook Pages. The pages of a notebook can be traversed by using `IPageHandle` objects, but the `INotebook` class provides a `Cursor` class that can do the job as well (and preserves the `Cursor` interface that is provided by other classes in the user interface class library). A `Cursor` object can be created by specifying the object the cursor will iterate over—in this case, the notebook itself. Then a `Cursor` method, like `setToFirst`, can be used to initialize the

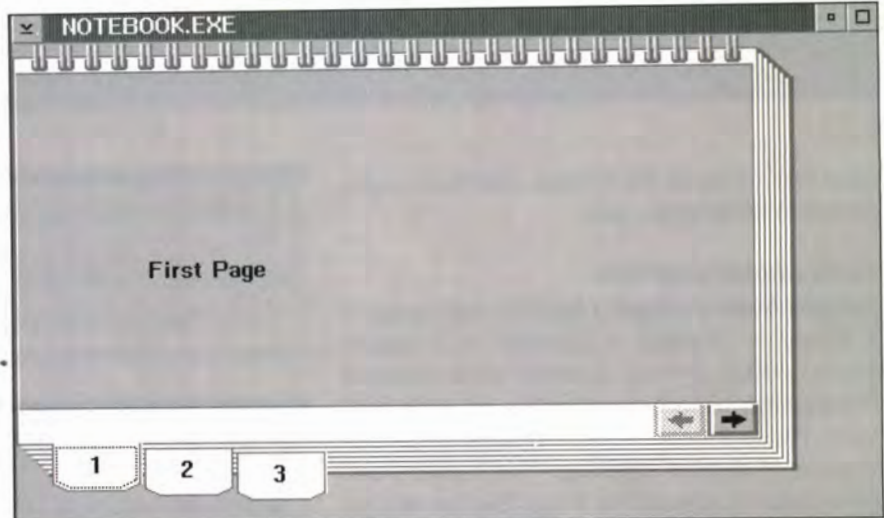


Figure 4. The notebook after three pages have been added

```
/* get a handle to the last notebook page and change its tab text */
IPageHandle lastPageHandle = pNotebook->lastPage();
pNotebook->setTabText (lastPageHandle, "Last");
```

Figure 5. Using the `IPageHandle` class

```
/* create a cursor for iterating over the pages of the notebook*/
INotebook::Cursor notebookCursor (*pNotebook);
for (notebookCursor.setToFirst(); // set the cursor to first page
     notebookCursor.isValid();    // keep going if cursor is valid
     notebookCursor.setToNext())  // move the cursor to next page
{
    /* turn to the page indicated by the notebook cursor */
    pNotebook->turnToPage (notebookCursor);
}
```

Figure 6. Using a notebook cursor

```
/* remove the page which the notebook indicates as its first page */
pNotebook->removePage (pNotebook->firstPage());
```

Figure 7. Removing a page

`Cursor` object to reference the first page of the notebook. As long as the cursor is in a valid state, it can be used to traverse the notebook pages one at a time. The code snippet in Figure 6 shows a `Cursor` object being created and used to display each notebook page in succession.

In Figure 6, the first line creates an

object of type `INotebook::Cursor`, which will iterate over the notebook pointed to by the `pNotebook` variable. In the for loop, the notebook cursor is initialized to the first page in the notebook with the `setToFirst` method. Each time through the loop, the `isValid` method is used to test the notebook cursor; when it has reached

OS/2 PRODUCTS & SERVICES

Opt-Tech Sort/Merge

High Performance Sort/Merge/Select Utility

Use as a stand-alone utility or call as a Subroutine. Many features including unlimited file size, multiple keys, record selection, duplicate elimination, summing and more!

Call for free brochure Opt-TechData Processing
P.O. Box 678
Available for: Zephyr Cove, NV 89448
OS/2 or Unix \$249 Phone (702) 588-3737
DOS or Windows \$149 Fax (702) 588-7576

Circle Reader Service Number 37

SUBSCRIBE TODAY!

Only \$39.95 for a full year subscription to the premier source of tools and techniques for the OS/2 software developer: **OS/2 DEVELOPER**

Call today and don't miss a single issue:

1-800-WANT-OS2

BUILD YOUR OWN OS/2 LIBRARY

Reprints are now available for **OS/2 DEVELOPER**.

Use your customized reprints for: **Customer support, sales tools, marketing support, educational tool.**

Call today for a custom quote: Laura Pullen, **OS/2 DEVELOPER**
415-358-0126 ext.302 or fax 415-358-9749

YOUR *single* SOURCE FOR OS/2 APPLICATIONS

UTILITIES	PRODUCTIVITY SOFTWARE
BOOKS & VIDEOS	GRAPHICS & CAD
OS/2 T-SHIRTS	COMMUNICATIONS
BACKUP SOFTWARE	NETWORKING
DEVELOPMENT TOOLS	WORD PROCESSING

ORDERS: 1-800-776-8284

FAX: 919-878-7479
INQUIRIES: 919-878-9700

3209 Gresham Lake Road, Suite 315 Raleigh, NC 27615 USA



Circle Reader Service Number 36

John C. Dvorak's Favorite

"The best of the best...Hobbes OS/2 CDROM... Lots of nifty utilities, fonts, and other goodies on this CD." -John Dvorak, PC Computing, Oct. 93.

Hobbes is 2800 files from the largest ftp archive of OS/2 material, ftp-os2.cdrom.com, quaintly known as Hobbes. Hobbes serves as the major internet distribution site of OS/2 freeware, shareware, device drivers, program updates and information. We bring this entire archive to you in the well organized, indexed, and virus scanned Hobbes disc. Many programs are installed on the CDROM, so you can run them without using any of your hard disk space.

The Hobbes OS/2 CDROM is updated quarterly and last updated in October 1993. Start enjoying this disc now. We guarantee your satisfaction.

\$29.95 Many more products available.

Walnut Creek CDROM • 4041 Pike Lane, Ste D-98 • Concord, CA 94520
1 800 786 9907

+1 510 674 0783 FAX: +1 510 674 0821 info@cdrom.com

Circle Reader Service Number 38

Attention Mathematica users:

Everything you need to know to get the most out of your math programming is in

THE MATHEMATICA JOURNAL

Call or fax today for information about subscriptions and special issues:

Phone orders: 415-905-2332

Fax orders: 415-905-2233

MISSING SOMETHING?

Complete back issue library available today!
Every back issue of OS/2 DEVELOPER can be yours.

Call today and complete your library:
1-800-WANT-OS2

IBM INTERNATIONAL SEMINAR

IS YOUR APPLICATION READY FOR THE GLOBAL MARKETPLACE?

- INSTRUCTOR TEAM FROM MAJOR IBM LABORATORIES, AND CONSULTANTS WITH YEARS OF EXPERIENCE
- DETAILED PROGRAMMING DESIGN GUIDELINES FOR INTERNATIONALIZATION
- NLS AND DBCS CONSIDERATIONS COVERED
- BREAKOUT SESSIONS FOR OS/2 WITH ACTUAL EXPERIENCES
- DEMOS, SAMPLE CODE, MANUALS, INDIVIDUAL ATTENTION

MARCH 1 AND 2, 1994
SANTA CLARA, CA MARRIOTT
\$500, INCLUDE MEALS

INFORMATION: 203-262-4034
INVITATIONS: FAX NAME &
ADDRESS TO
203-262-2141

FOLLOW-ON: OPTIONAL HANDS-ON PROGRAMMING LAB - FEE \$450

Circle Reader Service Number 35

the end of the notebook, the notebook cursor will no longer be valid, and the `isValid` test will halt the loop. After each loop iteration, the `setToNext` method causes the notebook cursor to reference the next page of the notebook. The `turnToPage` method used in the body of the loop causes the notebook to bring a specified page to the top, so that it is displayed on the screen. Either an `IPageHandle` object or an `INotebook::Cursor` object can be used as the argument for `turnToPage`.

Since `INotebook::Cursor` objects deal with `IPageHandle` objects, it is very easy to obtain one from the other. The `INotebook::Cursor` class returns `IPageHandle` objects from its `first`, `last`, `next`, `previous`, and `current` methods. An `IPageHandle` object can be sent to a notebook cursor with the `setCurrent` method to make a cursor reference the same page as the `IPageHandle` object.

Removing Notebook Pages. Once the notebook has been filled with pages, eventually some (or all) of those pages will need to be removed. The `INotebook` class provides several methods for removing pages, either one at a time or in groups. The `removeAllPages` method is self-explanatory, and the `removePage` method can take either an `INotebook::Cursor` object or an `IPageHandle` object as a reference to the page to be removed. The `removeTabSection` method also can use either an `INotebook::Cursor` or `IPageHandle` object, but its behavior is more complex. If the referenced page has a major tab, then that page and all following pages up to the next major tab page will be deleted. If the referenced page has a minor tab, then that page and all subsequent pages up to the next page with any tab will be removed. (If the referenced page has no tab at all, then no pages are deleted.)

An `IPageHandle` object referencing the first page of the notebook is implicitly created and used to delete the notebook's first page in Figure 7's line of code.

Counting Notebook Pages. The `INotebook` class also provides methods for

```

/*****
/* Class definition of MainWindow. It is both an IFrameWindow and an IPage-
Handler (a frame window that handles page events) */
*****/
class MainWindow : public IFrameWindow, public IPageHandler
{
public:
    MainWindow(unsigned long windowId);
    ~MainWindow ();

protected:
    /* these are IPageHandler events that MainWindow is overriding */
    virtual Boolean select ( IPageSelectEvent &event );
    virtual Boolean drawTab ( INotebookDrawItemEvent &event );

private:
    INotebook* pNotebook;
    IFrameWindow* pDialogs [3];
};

/*****
/* MainWindow constructor
*****/
MainWindow :: MainWindow (unsigned long windowId)
: IFrameWindow (windowId)
{
    IWindow* pOwner = this;
    IWindow* pParent = this;

    /* create a new INotebook object and save a pointer to it */
    pNotebook = new INotebook(ID_NOTEBOOK, pParent, pOwner);

    /* set up MainWindow as a handler of page events */
    IPageHandler::handleEventsFor (pNotebook);

    /* set the binding, tab shape, and orientation of the notebook */
    pNotebook->setBinding (INotebook::spiral);
    pNotebook->setTabShape (INotebook::polygon);
    pNotebook->setOrientation (INotebook::backpagesRightTabsBottom);

    long i = 0;
    /* create an IPageSettings object with the page-sizing and */
    /* major tab attributes */
    INotebook::PageSettings pageSettings
        (INotebook::PageSettings::autoPageSize
         INotebook::PageSettings::majorTab
        );
    for (; i < 3; i++)
    {
        pDialogs [i] = new IFrameWindow (ID_FIRSTPAGE + i); //load the dialog
        pageSettings.setTabText (IString (i + 1)); //set tab text
    }
}

```

Figure 8. Skeleton class definition, constructor, and destructor for a window that acts as a page handler (continued on page 60)

10 Reasons to Become a CSI Member

**BECOME
A CSI
MEMBER
AND
TAKE
ADVANTAGE
OF ALL
THESE
BENEFITS**

1. The Computer Security Alert—

CSI's monthly members-only newsletter that provides timely news and practical insights you can put to use immediately.

2. The Computer Security Journal—

The semi-annual journal of comprehensive, practical articles, case studies, reviews and commentaries, written by knowledgeable computer security pros covering the full spectrum of information security topics.

3. Educational Discounts—Members receive special discounts and benefits on CSI conferences, one- and two-day seminars and training.

4. Members' Hotline—For security emergencies, problems or questions, the CSI Members' Hotline is your direct connection to practical advice, resources and referrals.

5. Networking Opportunities—

Meet experienced colleagues who have already encountered and solved some of today's toughest security problems. Many CSI members report that networking is the #1 benefit of their membership.

6. Career Enhancement—As a CSI member, you have the most up-to-date information on computer security at your fingertips. CSI is a solid support resource you can count on—giving you expert, practical advice and helping you build your individual successes.

7. The Annual Computer Security

Buyers Guide—The most comprehensive resource available on computer security products and services.

8. On-line Bulletin Board—Our BBS

lets you access the latest computer security news, product announcements, industry alerts, technical updates and job listings.

9. The Manager's Guide Series—Five

copies each of these timely, non-technical explanations of computer security topics that include VirusThreats, Security Awareness, Telecommunications Fraud, Privacy and Encryption.

10. Publications Discounts—

Receive savings on books including The Computer Security Handbook & many other publications.



600 Harrison Street
San Francisco, CA 94107

Phone: (415) 905-2626
FAX: (415) 905-2218



using information about the number of pages in the notebook. A Boolean value is returned from the `isEmpty` method, which indicates whether or not the notebook has any pages in it. The `totalPages` and `pagesToEnd` methods return the number of total pages in the notebook and the number of pages from a specified page (again, using either an `IPageHandle` object or an `INotebook::Cursor` object) to the end of the notebook (including the specified page).

The `pagesToMajorTab` and `pagesToMinorTab` methods also can use either an `IPageHandle` or an `INotebook::Cursor` as an argument, but these methods only return the number of pages between the specified page up and the next page with a major tab or minor tab, respectively.

MORE ADVANCED TOPICS

Handling Notebook Events. The user interface class library has an extensive system of events and event handlers (the objects that handle the various events). Included in this architecture are several events that are particular to the notebook. These are called page events, and they notify their handlers of events like notebook pages being selected or deleted, the notebook window being resized, help being requested on a page tab, and a page tab being drawn. In order to be notified of these events, developers must use the `IPageHandler` class (sometimes the owner window of the notebook is used as a subclass of `IPageHandler`).

Suppose, for example, a notebook should beep whenever the last page of the notebook is displayed. If the notebook is the client area of the main window, then the main window could be a subclass of `IPageHandler` as well as `IFrameWindow` (and whatever other classes or handlers it is using). After the notebook has been created and inserted into the client area (usually by the `IFrameWindow` method, `setClient`), the main window can use the `IPageHandler::handleEventsFor` method to indicate that it will be handling page

```
pNotebook->addLastPage (pageSettings, pDialogs [i]); //add to notebook
}

/* make the notebook the client area of MainWindow */
setClient(pNotebook);

setFocus();
show();
}

/*****
/* MainWindow destructor
*****/
MainWindow::~MainWindow ()
{
/* stop handling the page events of the notebook */
IPageHandler::stopHandlingEventsFor (pNotebook);
long i = 0;
for (; i < 3; i++)
{
delete (pDialogs [i]); // free space allocated for dialogs
}
delete pNotebook; // free space allocated for notebook}
}
```

Figure 8. Skeleton class definition, constructor, and destructor for a window that acts as a page handler (continued from page 58)

events. As a result of this call, particular methods in the `IPageHandler` class will be called using `IPageEvent` objects as parameters. The main window can then intercept individual events by defining and implementing those methods (since the main window is an `IPageHandler` subclass). Similarly, when the main window no longer needs to intercept page events, it should call the `IPageHandler::stopHandlingEventsFor` method. Figure 8 shows a skeleton class definition, constructor, and destructor for a main window that acts as a page event handler and uses page events to generate the last page beep.

The class definition defines `MainWindow` as a subclass of both `IFrameWindow` and `IPageHandler`. In its constructor, `MainWindow` creates a notebook (`pNotebook`, private `MainWindow` data, is a pointer to an `INotebook` object), adds pages, and uses `setClient` (an `IFrameWindow` method) to put the `INotebook` object in its client area. `MainWindow`

uses the `IPageHandler::handleEventsFor` method to establish itself as a handler of the page events generated by the notebook object referenced by `pNotebook`. Anytime a page event is generated by that notebook object, the page event will be sent to `MainWindow`, and any page events that `MainWindow` does not handle will be passed to the default handler, the `IPageHandler` class itself.

Also in the `MainWindow` class definition are declarations of the `IPageHandler` methods, which `MainWindow` intends to override. These methods are listed in the protected section of the `MainWindow` class definition and have Boolean return codes; the Boolean values returned from these methods indicate whether processing of the event should stop with this handler (true) or whether processing should continue with any other handlers registered (false). The `select` method is called when a notebook page is selected.



HARNESS THE POWER OF OS/2

INTRODUCING OS/2 MAGAZINE



Make the most of your PC with *OS/2 Magazine*! Every issue brings you the information you need to achieve maximum performance. From your hardware and software, to your operating system and peripherals.

OS/2 Magazine helps you harness the power of your PC. Why not flip open the next issue to timely feature articles like these:

- Common OS/2 problems and how to solve them
- How to detect OS/2 viruses—and keep your data safe
- Putting your OS/2 PC onto a local-area network
- Making multimedia applications come alive under OS/2
- Running Windows applications under the Workplace Shell

You'll also get hands-on tutorials. Expert technical advice. And unequivocally, unbiased product reviews—on HP LaserJets, S3 graphic boards and Intel fax modems, to name a few. Plus the latest news on OS/2, invaluable network support and more.

SEE FOR YOURSELF

First issue FREE! Check out the next issue of *OS/2 Magazine*. It's on us. If you like it, you can get a full year's worth (11 additional issues) for the low charter rate of \$29.95. You'll save 25% off the regular subscription rate!

Why not return the attached form today? You have so much to gain, and *absolutely* nothing to lose!

**SPECIAL
INTRODUCTORY
OFFER!**



YES! Send me a complimentary copy of *OS/2 Magazine*. If I like it, I can get a full year's worth (11 additional issues) for the low charter rate of \$29.95. I save 25% off what regular subscribers pay. If *OS/2 Magazine* is not for me, I'll simply write "cancel" on your invoice and that will be that. I owe *nothing*. I lose *nothing*. And the FREE issue is mine to keep.

NAME _____

ADDRESS _____

CITY/STATE/ZIP _____

Mail your order to: *OS/2 Magazine*, P.O. Box 56664, Boulder, CO 80323-6664, or call (800) 765-1291, or Fax (415) 905-2233.

Allow up to 6-8 weeks for delivery of your first issue. Canadian/Mexican subscribers please add \$10 for surface mail; overseas add \$40 for airmail. Prepayment drawn in U.S. dollars on a U.S. bank for foreign orders.

In turn, the `MainWindow` destructor must call the `IPageHandler::stopHandlingEventsFor` method so that `MainWindow` will stop receiving page events generated by the notebook object. (In this example, identifying `handleEventsFor` and `stopHandlingEventsFor` as `IPageHandler` methods is not really necessary, since `MainWindow` doesn't subclass any other handlers. However, if `MainWindow` did subclass other handlers, it is much clearer to identify which versions of `handleEventsFor` and `stopHandlingEventsFor` are being called.)

The `MainWindow::select` method overrides the default handling of the page selection event provided in the `IPageHandler` superclass. The `IPageSelectEvent` object that is passed to the select method has methods of its own that can provide `IPageHandle` objects referencing the selected notebook page or the previously selected notebook page as well as a pointer to the `INotebook` instance where the selection event was generated. In the example, the `pageHandle` method is invoked to get a reference to the selected page, and that `IPageHandle` is compared against the `IPageHandle` returned from `INotebook's lastPage` method to determine if the last page of the notebook is the selected page. If so, a beep is generated. Finally, false is returned to signal that processing should continue with any other handlers. This is a very simple use of the select method, but with the various pieces of information provided by the `IPageSelectEvent` object, it's easy to see how simple it is to implement various behaviors like changing the tab text status text of selected pages, or even displaying a page other than the one selected! The other page events—like `help`, `remove`, and `resize`—all work similarly.

Drawing Notebook Tabs. A fun event to handle is the drawing of notebook tabs rather than defining text or bitmaps for them. Handling tab-drawing events is similar to handling other page events, although some different information is required from the `INotebookDrawItemEvent`

```
Boolean MainWindow::drawTab (INotebookDrawItemEvent& event)
/*****
/* This method will be called when a page tab is to be drawn. */
*****/
{
    POINTL pCenter = event.itemRect().center().asPOINTL(); // rect center
    ARCPARAMS arcp = {1, 1, 0, 0}; // arc parameters
    HPS hps = (HPS) event.itemPresSpaceHandle(); // presentation space
    long j, i,
        clrArray [4] = {CLR_BLUE, CLR_RED, CLR_GREEN, CLR_YELLOW};

    GpiSetCurrentPosition (hps, &pCenter);
    GpiSetArcParams (hps, &arcp);
    for (i = 0, j = 20; j >= 5; j -= 5, i++)
    {
        GpiSetColor (hps, clrArray [i]); // change color
        GpiFullArc (hps, DRO_OUTLINEFILL, MAKEFIXED (j, 0)); // draw circle
    }
    return (true); // no more processing of this event
}
```

Figure 9. Handling the tab-drawing event

`mEvent` object that is passed on the `drawTab` method.

Like the other page event classes, `INotebookDrawItemEvent` has a method for returning an `IPageHandle` object referencing the page involved in the event (in this case, the page with the tab to be drawn). In addition, `INotebookDrawItemEvent` inherits other methods from its superclass, `IDrawItemEvent`, for obtaining the presentation space

to use when drawing and the rectangle that bounds the tab's drawing area. The example in Figure 9 shows some simple drawing in page tabs.

In the Figure 9 sample, the `itemPresSpaceHandle` and `itemRect` methods that `INotebookDrawItemEvent` inherits from `IDrawItemEvent` are used to obtain, respectively, the presentation space for drawing and the rectangle in which to draw (the rectangle rep-

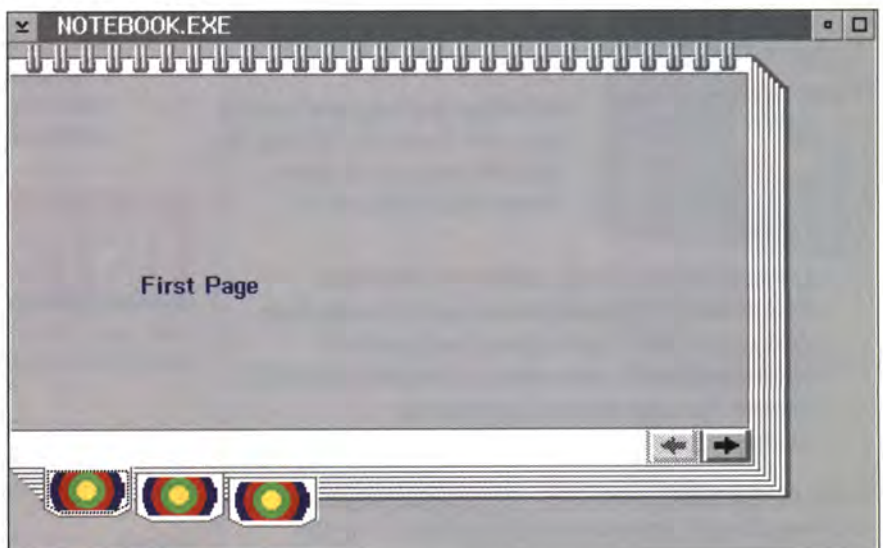
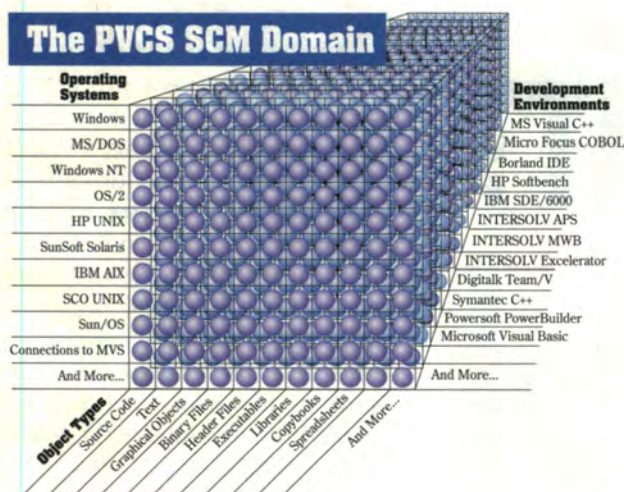


Figure 10. The notebook after the tab drawings



"Our Client/Server tool vendor recommended PVCS to keep development on track. When we bought PVCS we were pleased to see it integrated into all the tools we use in all the environments we support. PVCS solved development problems that we had been working around for years. The productivity and quality boost is huge."

For Productive LAN-based Development, Software Configuration Management is Not Just Nice to Have...



PVCS gives you a complete SCM solution that wraps around all of the popular operating systems, development environments and object types. In an increasingly heterogeneous development world, PVCS allows you to avoid fragmenting your culture with different SCM mechanisms on every desktop.

**Moving
SCM
to the
LAN**

...It's a Necessity

You can't afford confusion. You can't afford changes being overwritten, or not knowing who made changes, what they were, and why they were made. You need to build systems with repeatable accuracy, automatically. You need to be able to document everything that gets done, without spending your valuable time doing housekeeping. You need PVCS.

You Need a Complete Solution

But the last thing anyone needs are important tools that only work in a limited environment. PVCS solves the software configuration management problem wherever you need it. It works the same in all environments.

10,000 Companies Just Like Yours

More than 10,000 companies worldwide have standardized on PVCS. There's only one reason why more people use PVCS than all other SCM products combined. It's the best solution. Call for more information.

**Call Today for
More Information:
800-547-PVCS Ext. 5**

i
INTERSOLV

The Standard for Software Configuration Management on the LAN

Inside U.S.A. • 1700 NW 167th Place • Beaverton, OR 97006 • 503-645-1150 • FAX 503-645-4576 • E-Mail PVCSINFO@INTERSOLV.COM
Outside U.S.A. • Abbey View • Everard Close • St. Albans • Herts AL1 2PS • United Kingdom • Tel 0727 812 812 • FAX +44 727 869 804

Circle Reader Service Number 46



REFERENCES

User Interface Class Library User's Guide (Mechanicsburg ordering number S71G-2230-00)

OS/2 2.0 Programming Guide Volume II (S10G-6494-00)

Presentation Manager Programming Reference Volume I (S10G-6264-00) and *Volume II* (S10G-6265-00)



Developing today's complex applications can be quite puzzling. It requires more and more knowledge and training. A complete understanding of the capabilities of OS/2 and the establishment of programming standards will help you produce applications that give your business a competitive edge in the market place. Ask the OS/2 specialists...

We can help you create the most productive environment for your application developers through training, consulting, and programming services. Our object-oriented software engineering and GUI design techniques are the keys to make your application development a success.

Benefit from our proven experience!



International, Inc.

621 NW 53rd Street, Suite 240
Boca Raton, FL 33487 USA
1-800-4SE-INTL or (407) 241-3428
Fax: (407) 278-3919

SES GmbH Berlin

Koenigsallee 49
14193 Berlin, Germany
+49 30 826 40 63
Fax: +49 30 825 30 14



resents the area of the page tab). The tab rectangle is used to find the center of the drawing area, and the presentation space handle is used to set the current position to the center of the drawing rectangle and then to draw overlapping circles on the tab. If different drawings are needed on different tabs, `INotebookDrawItemEvent`'s `page` method can be used to find out which page's tab is being drawn. Finally, `true` is returned to indicate that this `IPageHandler` object has done the drawing, and the draw tab event does not need to be processed by any other handlers.

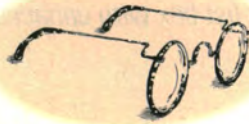
Figure 10 contains the results results of the tab drawings.

CONCLUSION

The interface to the notebook control provided by the user interface class library is very powerful and comprehensive. Familiar Presentation Manager functions and styles are fully available, and event processing is simple to implement through the user interface class library's extensive handler-and-event architecture. The behavior of the `INotebook` object and related objects can even be extended through the subclassing and overriding capabilities of the C Set ++ compiler. In summary, the notebook control of the user interface class library is just as powerful—and even more extendible—than under Presentation Manager alone.

Brad Broyles, IBM Store Systems, Raleigh, NC, is a senior associate programmer in Future Applications Development. He joined IBM at the Cary Software Development Laboratory in 1988, where he contributed to the IBM SAA Common User Access Controls Library/2 and the User Interface Class Library, among other projects. Broyles moved to Store Systems in 1993 as a user interface developer. He received a B.S. in computer science and mathematics from the University of Tennessee, Knoxville and is currently working towards an M.B.A. from the University of North Carolina at Chapel Hill.

{ figure 1. }



SPECTACLES

{ figure 2. }



OBSERVATORY

Is Your Vision In Focus?

If you're like most development managers, your vision is to build the best software possible for your company. The challenge is to focus on which new technologies offer the most immediate payback to your development objectives.

With OOP, code reusability is more of a reality. Visual tools offer shortcuts in the development cycle. Distributed architectures and GUIs offer better user access to information. Whichever of these you've decided to focus on this year, you'll find help turning your vision into reality at **Software Development '94** and **Business Software Solutions**.

Together for the first time this March, these two important industry events join forces to bring you the most extensive educational opportunity ever assembled for software development teams.



In over 200 courses you'll get hands-on help from experts with everything from programming in C++, Windows development, and GUI design, to moving your mainframe development staff to client/server, to building better information delivery systems with today's high-end tools. You'll also witness 250 vendors (like Microsoft, CA, IBM, KnowledgeWare, Borland, Gupta, Novell) race to bring you the best support for your efforts at the biggest development tools exhibition in the country.

If you're focused on building the best software for your company, take steps now to make your vision a reality by attending **SD '94 / Business Software Solutions**.

200 classes • 250 exhibitors • 25,000 of your peers



CONFERENCE & EXHIBITION

March 14-18, 1993
San Jose Convention Center

Call us at
415-905-2784
to find out more.



Circle Reader Service Number 48



GUI

This is the first of a series of articles in which we develop a replacement list box with enhanced capabilities.

BY MARK A. BENGE and MATT SMITH

A List Box Replacement



Mark A. Bengé



Matt Smith

This month, we will explain how to write a replacement of the OS/2 Presentation Manager list box. This replacement will duplicate all of the existing functionality, so you can use it in place of the current list box. One very significant enhancement will be the removal of the memory limitation that plagues the current list box. We cannot show the entire source code for the list box here due to space limitations, but you can obtain it from resources listed in the References section at the end of the article.

The current, list-box design within OS/2 Presentation Manager has been around since OS/2 1.1 with very little enhancement. The first new feature added was the horizontal scroll bar in OS/2 1.2. The only new feature added to it after that occurred in OS/2 2.0, in which the extended-selection capabilities were allowed to be specified as the style `LS_EXTENDEDSEL`. This list box is old in terms of control design; it provides no control data, has data limitations, and is just quirky.

Like most of you, we have encountered the list-box limitations many times. Our patience ran out, and we were tired of writing code to get around some of the limitations, so we wrote a replacement for the list box. But not just any list box. We needed a list box that could handle more than 2,000 items or 64K data (whichever the control decided to penalize you with), is fast, and can put the cat out when needed.

WHEN TO SUBCLASS

One way of updating the list box is by subclassing. We considered this option very briefly. With sub-

classing, you can achieve some of the design goals we will cover here, but you would most definitely be hindered by the memory constraints, painting, searching, and sorting. We needed total control over the list box if we were to get the desired performance. The major drawback is that we have to do all the work; we can't assign anything to an existing control.

LIST-BOX DESIGN

Visually, the list box is not overly complicated, but our added enhancements will make it programmatically complex. As we add new capabilities, the design will evolve. We will show you how to design controls that take into consideration existing versions yet allow for increased functionality.

The existing list box from OS/2 Presentation Manager consists of three, distinct visual components. The main component, the list area represented by the ID of the control, creates and maintains two additional components: the vertical and horizontal scroll bars, having the IDs of `0xc001` and `0xc002`, respectively. Figure 1 shows the IDs of the two scroll bars.

Our first goal is to maintain full compatibility with the existing OS/2 Presentation Manager list box. Therefore, our first version of the list box will handle all the styles listed in Table 1 and the messages in Table 2. In addition, we will duplicate the existing mouse and keyboard handling.

How does this translate into a control design? Basically, we will divide our control into the four areas shown in Figure 2. The heap manager, an important area, is required since we can't use the

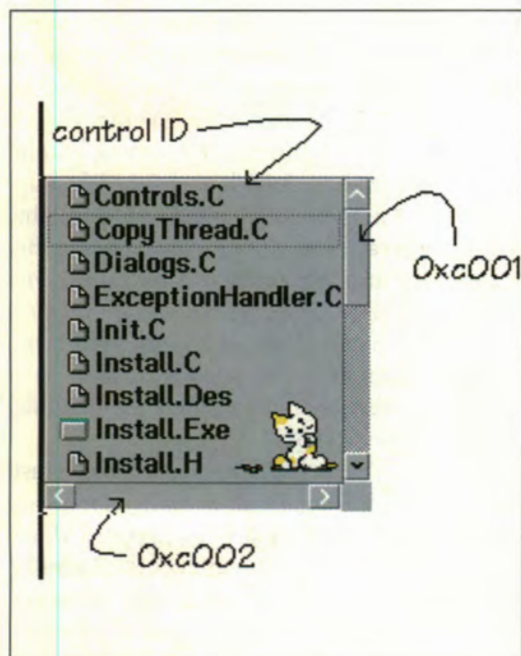


Figure 1. Existing list box design

heap management designed for use by controls of OS/2 Presentation Manager. The heap management of PM places a limit of 64K available for use by each window or control. This heap management is what causes everyone to run into a brick wall. Later, we will explain in detail why we have to implement our own memory-management scheme and forego the memory-management services provided by the compiler.

The second area, the list manager, provides the means for manipulating the information contained within the list box. It will manage the insertion, appending, or removal of list items.

The event manager is used to handle the mouse and keyboard-control notification. It ensures that items are correctly highlighted, displayed, and based on the user's selections.

The final area, the traffic manager, is used, essentially, to direct traffic. Depending on the message from the user, the traffic manager will give us the means of providing information on the list box.

ENHANCING THE LIST-BOX DESIGN

Some of our enhancements to the list box (which will be discussed in later articles) are column support, block list setting, delayed sorting, graphic support, direct editing, drag and drop, dynamic data exchange, variable CTLDATA, and enhanced display. Additional enhancements will be added as they become significant.

Consequently, we have to plan for some of these issues when we create our first version of the

Style	Value	Meaning
LS_EXTENDEDSEL	0x00000010L	Allows extended selection interface of more than one item at a time within the list box.
LS_HORZSCROLL	0x00000008L	Provides horizontal scroll bar at the bottom of the list box.
LS_MULTIPLESEL	0x00000001L	Allows selection of more than one item at a time within the list box.
LS_NOADJUSTPOS	0x00000004L	Causes the list box to be drawn using the size provided.
LS_OWNERDRAW	0x00000002L	Owner of the list box draws each item within the list box if desired.

Table 1. Existing list box styles

Style	Value	Meaning
LM_DELETEALL	0x016e	Deletes all items in list-box control.
LM_DELETEITEM	0x0163	Deletes a specified item from the list.
LM_INSERTITEM	0x0161	Inserts item text in a list-box.
LM_QUERYITEMCOUNT	0x0160	Retrieves number of items in a list-box.
LM_QUERYITEMHANDLE	0x016a	Retrieves list-box item's handle.
LM_QUERYITEMTEXT	0x0168	Copies text from item in list-box.
LM_QUERYITEMTEXTLENGTH	0x0167	Retrieves item's text length in list-box.
LM_QUERYSELECTION	0x0165	Retrieves index of selected item in list-box.
LM_QUERYTOPINDEX	0x016d	Retrieves index of top item in list-box.
LM_SEARCHSTRING	0x016b	Searches list-box for a match.
LM_SELECTITEM	0x0164	Sets an item's selection state in list-box.
LM_SETITEMHANDLE	0x0169	Sets an item's handle in list-box.
LM_SETITEMHEIGHT	0x016c	Sets height of items in a list-box.
LM_SETITEMTEXT	0x0166	Sets item text in a list-box.
LM_SETTOPINDEX	0x0162	Sets item to top of list box.

Table 2. Existing list box messages



list box. The most significant feature is the column support. This affects the manner in which we store the data submitted to the list box.

Under the current design, we only

of manipulating the information. For example, if we use a set of link lists for the rows and columns, inserting and deleting can become quite interesting. If we use the array method, we either

these will be implemented at the same time. In fact, some of the features found in the messages will be implemented in this first version of the list box.

One feature is data sorting within the list box. A trick we have picked up is the hiding of the list box using the `WinEnableWindowUpdate` call, thus forcing the list box to hide itself effectively while we insert items within it. Once we have placed the items within it, either sorted or unsorted, we issue a `WinShowWindow` call to force the painting of the list box.

What is really happening here? Just a conjuring trick, actually. First, we must understand what happens when we don't use the `WinEnableWindowUpdate` call. When we send the message `LM_INSERTITEM` to the list box, the list-box code adds the item in `mp2` to its internal list. Using the value in `mp1`, it either leaves the item at the end of the list (`LIT_END`), sorts the list in ascending order (`LIT_SORT-ASCENDING`) or descending order (`LIT_SORTDESCENDING`), or places it in the position specified in `mp1`.

As soon as the item has been processed, the scroll bar for the list box is updated (which may force a repainting of it) and, depending on the `mp1` value, may cause the contents of the list box itself to be repainted. How badly the forced repainting is depends on the sorting characteristics and the data being sorted.

Interestingly enough, the current OS/2 Presentation Manager list box doesn't have the intelligence to speed-up the fill operation, since it doesn't pay attention to its visibility state. Since we are not going to use linked lists for our list box, we will check to see if the list box is visible when a new item is inserted. We will set a flag within the internal data so that when the list box is shown, we will sort the information within it before the list box is painted. Likewise, if the sorting characteristic is changed, for example from `LIT_SORTASCENDING` to `LIT_END`, we would perform the sort at that time as well.

This delayed sorting will correspond to the `LMX_SORT` message, which we will add to the list box in a later article.

So What's Wrong with the C Run-time Memory Management?

Nothing, really—if you are writing most applications. But if you are writing a control where you need to allocate and release memory quite often, the C run-time memory management is most definitely not the thing to use. One of the reasons is that it is unforgiving. We all make mistakes, since we are not perfect (except for maybe Dr. K), but the C run-time libraries expect us to be. If you forget to release a chunk of memory you have requested, the run-time memory-managers routines won't do it for you since they have to assume (and rightly so) that the memory is still being used. Also, you are at the mercy of the library writer as to how efficient the routines are with the memory. You may be quite surprised how much memory is allocated for that chunk of memory you requested. Almost all of the C run-time uses the same premise of a static block which starts the memory chain of a larger block of memory that is then sub-divided through the `malloc` and `calloc` functions. Larger requests for memory (32K or 64K and larger) are sometimes allocated from the system directly, such that the larger block is not part of a linked list. You are expected to free this properly, or otherwise, in the case of a DLL, it may just decide to stick around. Remember, memory is allocated on a per process basis, therefore a piece of memory allocated in a dynamically loaded DLL will stick around even after the DLL has been removed from memory.

have the capability to look at the information as a one-column-by-many-row design. Therefore, we can delete by individual row or by the entire list box (the one column).

With a multicolumn design, we should be able to delete in one of three ways—the two methods of the original design and the third, by column.

This feature affects how we store the items within a list box. For the new list-box design, we can store the items of the list box as an array of items or as a linked list. Both have their advantages and disadvantages.

For the multicolumn list box, both can be used. However, before any code is written, we must consider the means

shuffle up or down the array elements when inserting or deleting. Row *x* is the same in all columns and each column can be an array in its own right.

This makes some of the design, in terms of the user interaction, interesting. The array approach means that the deletion of a column or row is quite difficult depending on what is being deleted (the column can be the fastest depending on memory management). The linked-list method means updating the pointers both horizontally and vertically. One mistake produces a big mess. In this case, we will use the array approach.

Table 3 shows some of the new styles planned, and Table 4 contains some of the new messages. Not all of

Depending on the number of items, this could speed up things considerably.

HEAP MANAGER

We will implement our own memory manager to control the memory usage within the list box. Presently, the OS/2 Presentation Manager list box borrows memory from its owner window through the owner-window heap.

Under OS/2 1.x, a message, `WM_CONTROLHEAP`, was used to allow the user to specify a different heap. This message allows each control a maximum of 64K if the owner of the control wants to provide it. Under OS/2 2.x, this message was removed.

Therefore, all controls must use the same 64K heap for memory, meaning that the list box we were using may or may not handle the same number of the items depending on the memory load of the owner window.

We can't use the C run-time library memory managers, since they do not allow us to control the manner in which memory is allocated. They also cannot guarantee minimal, memory configurations when we absolutely require it.

What we are going to do here is use a heap-based system that does not require a fixed starting block. In place of the fixed starting block, we will use a dynamic starting block, which we will use as a handle to each different heap. Therefore, we can design our control so that it uses different heaps for different things. And, due to this design, we can quickly discard a heap, ensuring that the memory is properly released back to the system. The goal is to make the memory management as fast as possible and guarantee that we can maintain minimal memory impact.

To make the transition from the traditional, memory-management calls of `malloc()`, `calloc()`, `realloc()`, and `free()` shown in Table 5, the memory management has duplicated the syntax of the calls mostly with the addition of a parameter. The first parameter in the call must be the heap handle.

Two additional calls must be used to initialize the heap and finally release the memory owned by the heap back

Style	Value	Meaning
LSX_AUTODISPLAY	0x00000100L	Allows the list box to remove or show the scroll bars depending on the contents of the list box.
LSX_LEFTVERTSCROLL	0x00000200L	Provides vertical scroll bar on the left size of the list box.
LSX_TOPHORZSCROLL	0x00000400L	Provides horizontal scroll bar on the top of the list box.
LSX_NORIGHTVERTSCROLL	0x00000800L	Removes the right vertical scrollbar.
LSX_COLUMNS	0x00001000L	Permits columns to be used within the list box.
LSX_EDITABLE	0x00002000L	Permits editing of items within the list box.
LSX_NOCUTPASTE	0x00004000L	Disables cutting and pasting within list box.

Table 3. Enhanced list box styles

Style	Value	Meaning
LMX_COPY	0x0170	Copy item to clipboard.
LMX_CUT	0x0171	Cut item to clipboard.
LMX_PASTE	0x0172	Paste item from clipboard.
LMX_QUERYITEMCOUNT	0x0173	Query list box count.
LMX_QUERYITEMRECT	0x0174	Query item display rectangle.
LMX_QUERYSELECTLIST	0x0175	Query multiple selection list.
LMX_RESETLIST	0x0176	Reset multiple selection list.
LMX_SELECTALL	0x0177	Sets all item's selection state in list-box to on.
LMX_SELECTITEMBITMAP	0x0178	Select item bitmap.
LMX_SETARRAY	0x0179	Sets list box list using an array of PSZ.
LMX_SETARRAYHANDLES	0x017a	Sets list box list of handles using an array of ULONG. Used mostly for OWNERDRAW.
LMX_SETITEMBITMAPS	0x017b	Set item bitmaps.
LMX_SETITEMCOUNT	0x017c	Set list box count.
LMX_SORT	0x017d	Sort list box.

Table 4. Enhanced list box messages.

HHEAPMEM	HeapAlloc(ULONG cbInitial, ULONG cbNewBlks);
PVOID	HeapCalloc(HHEAPMEM hHeap, ULONG cItems, ULONG cbSize);
PVOID	HeapMalloc(HHEAPMEM hHeap, ULONG cbSize);
PVOID	HeapRealloc(HHEAPMEM hHeap, PVOID pv, ULONG cbSize);
ULONG	HeapSize(HHEAPMEM hHeap);
VOID	HeapDisplayStatus(HHEAPMEM hHeap);
VOID	HeapFree(HHEAPMEM hHeap, PVOID pv);
VOID	HeapRelease(HHEAPMEM hHeap);
VOID	HeapStatus(HHEAPMEM hHeap, PULONG pcBlocks, PULONG puLSize, PULONG puLUsed, PULONG puLFree, PULONG puLUnused, PULONG puLOverhead);

Table 5. Heap manager functions

to the system. The `HeapAlloc` call starts the heap off by requesting a block of memory using the `DosAllocMem` system call. The address returned by `DosAllocMem` is the starting block for the heap, and this value is used as the heap handle. This value, when passed to the `HeapMalloc`, `HeapCalloc`, `HeapRealloc`, and `HeapFree` routines, allows each to select the starting block correctly and determine from which memory chain to carve or free the memory.

When the heap is to be discarded after it is no longer needed, the `HeapRelease` call is used. Depending on the circumstances, this may be the only call required, since all of the memory of the heap is released back to the system. There is no need to release individual memory blocks for the heap, since the memory blocks that have been carved up from the heap will be discarded.

**So many
industry leaders
agree on
one object database,**

**isn't it worth
a phone call?
1-800-962-9620**

It sure is. Because when you call Object Design®, you'll get the inside story on the object database from a company Ray Noorda called "*the driving force behind the future of ODBMS,*" Scott McNealy described as "*the foundation for a new generation of enterprise-wide applications*" and Steve Mills declared is "*the leader in object database technology.*"

It's called ObjectStore™, and discovering what these leaders already know is easy. Just call for your free Object Database Evaluation Kit, featuring an Evaluation Guide, user stories and our video.

With the Kit, you'll learn what to look for in an object database. You'll see the importance of interactive performance, scalability and persistence independent of type. And you'll discover how our innovative virtual memory mapping architecture delivers those benefits, making ObjectStore the world's leading object database.

It's all in the Kit. Along with details on exclusive features. So if you're ready to choose an object database, call for your Evaluation Kit now. And see what everyone is talking – and agreeing – about.



Get your free Kit now!

**1-800
962-9620**
Or e-mail
info@odi.com


OBJECT DESIGN
Leadership by Design®

Object Design, Inc.
Corporate Headquarters
Burlington, MA USA
Wiesbaden, Germany: 49-611-39707-0
Swindon, UK: 44-793-486111
Sydney, Australia: 61-2-212-2766
Tokyo, Japan: 81-3-3251-2882

© 1993 Object Design, Inc. Object Design and Leadership by Design are registered trademarks and ObjectStore is a trademark of Object Design, Inc.

At this

So are lava lamps and bell-bottoms.

The theme of our conference is "retro," but the technologies are up-to-the-minute. That's why

TECHNICAL



SAN FRANCISCO 94

software designers, developers, technical coordinators, device driver developers, LAN specialists, MIS managers, consultants and training executives definitely

shouldn't miss it.

Improve your productivity with sessions on OS/2®, LAN systems, graphics, OOP, multimedia, pen, database and communications. For device driver developers, special seminars on displays, printers, storage, LANs and input devices are offered.

Everyone is invited to experiment in our OS/2

bugs are



software conference,

and LAN lab, and see exhibits from software vendors who exploit PSP products.

See the latest PSP technologies, April 25-29, 1994
Now including the Device Driver Conference

Speakers include IBM's David Proctor, VP of the PSP Division, and John Soyring, Director of Software Development Programs, with an industry keynote address by Computer Associates International, Inc. Chairman and CEO Charles B. Wang. Also, enjoy premiums, raffles, a "special event" and

more. Register before March 21, 1994 and save \$100 on the \$895 fee. For more information or to sign up, call 1 800 872-7109 (USA and Canada)*.

It's gonna be groovy.

Operate at a higher level.™

*Outside the USA and Canada, call 1 508 443-4990. Volkswagen, the Volkswagen logo and "Beetle" are registered trademarks of Volkswagen, AG. Used by permission of Volkswagen, AG. IBM and OS/2 are registered trademarks and "Operate at a higher level" is a trademark of International Business Machines Corporation. © 1993 IBM Corp.



Circle Reader Service Number 52

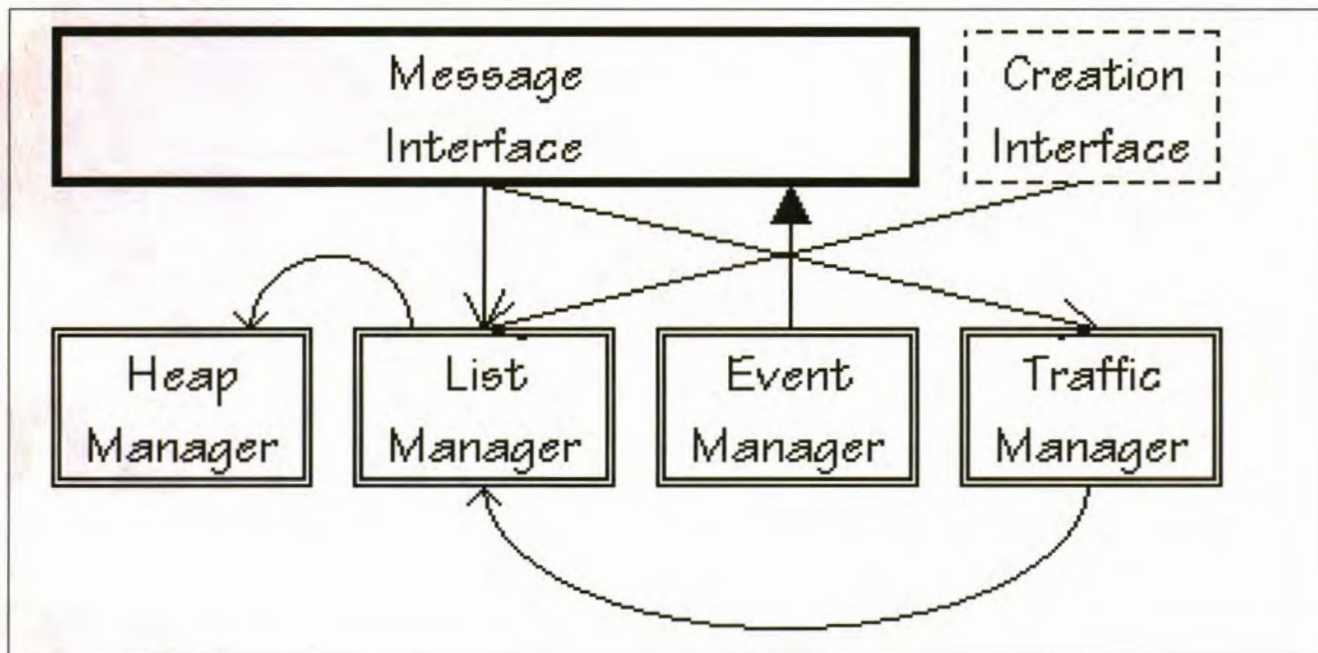


Figure 2. Control structure

LIST MANAGER

The list manager of the control provides the main interface to the user of the control. The messages of Tables 2 and 4 go through either the list-management area or the traffic-management area, depending on the action requested.

List management deals with insert, delete, and query of list-box items. It should be designed so that it allows control data to be incorporated.

This control data will take the form of information from some of the messages in Tables 2 and 4 (for example, LMX_SETITEMCOUNT) and for the setting of the data within the list.

EVENT MANAGER

The event management provides the reverse of the list and traffic managers; the control sends information back to the owner, when significant events occur. This is generally denoted by WM_CONTROL messages, which indicate a scrolling action, item selection, deselection, and so on.

TRAFFIC CONTROLLER

The traffic controller handles odd-ball requests from the user, which don't directly manipulate the contents of the list box. Messages, such as LM_QUERYTOPINDEX and LM_SETTOPINDEX, are found within the traffic-management area. It may communicate with the list-box-

management code but, in general, does not affect the contents of the lists. In other words, it is passive in nature.

BEYOND THE BASICS

Next time, we will explain more of the new list box design and begin the task of adding some of the new features. As we progress on this journey, we will keep adding more features until we have a list box that really illustrates the power underneath the PM hood.

Mark A. Benge, IBM Programming System Laboratory, Cary, N.C., is a staff programmer who joined IBM in 1989 and has worked on various CUA '91 controls for OS/2 2.x and CUA Controls Library/2. He works in IBM user interface class library development, where he was involved in the implementation of C++ classes for direct manipulation for the C Set++ 2.1 product. Benge has a B.S. in computer science from Western Carolina University.

Matt Smith, Prominare Inc., Toronto, Ont., Canada, is lead architect for the Prominare Development System, an OS/2 2.x advanced GUI development environment. He has been actively involved with OS/2 since 1988 and co-founded Prominare in 1990. Smith has a degree in architecture from the University of Waterloo.

REFERENCES

The source code can be downloaded from several electronic sources: Call the IBM PCC BBS at (919) 517-0001. The source code is in File Area 11 under the name LSTBX1.EXE.

On CompuServe, the source code is in LSTBX1.EXE of the OS2DF2 forum, OS/2 Developer library.

If you have a VM account on an IBM node, you can receive the source code with the command REQUEST LSTBX1.EXE FROM BANZAI AT CARVM3.



You can easily write additional functions to extend the function base available to the REXX programmer. This article provides an explanation and source code for a simple set of external functions written in C.

By ANDREI MALACINSKI and PATRICK MUELLER

Extending REXX with External Functions

The REXX language, provided as part of OS/2 2.0 and 2.1, is an easy-to-learn and powerful interpreted language. Besides offering a number of built-in functions, it allows users to add functions to the language by writing external functions in C. These external functions can be written to enhance the function set provided or to become small layers of code on top of a set of existing C functions, making a C function package available in REXX.

This article provides the source for a simple set of external functions written in C. The article notes differences between the C programming environment and the REXX programming environment and provides hints and pitfalls to avoid when writing external functions.

EXTERNAL FUNCTIONS

REXX contains a number of useful built-in functions. REXXUTIL.DLL, an external function package provided with OS/2, contains additional OS/2-related functions. Here, we describe how you can create external functions packaged in dynamic link libraries (DLLs), such as REXXUTIL.DLL.

The external functions that make up the DLL are written in C with a special interface that makes them callable from REXX. Each REXX function you provide needs a C function that implements the REXX function. More than one REXX function can use the same C function.

In C, a function prototype can have many formats: multiple parameters, call by reference, and call by value. C functions that implement REXX external functions must have the same prototype.

The prototype must correspond to the following:

```
ULONG APIENTRY FunctionName(  
    PCHAR    pszName,  
    ULONG    ulArgc,  
    PRXSTRING prxArgv,  
    PSZ      pszQueueName,  
    PRXSTRING prxRet  
)
```

The arguments are:

- **pszName.** This is the name of the REXX function invoked. When more than one REXX function is implemented by the same C function, this parameter is used to determine which REXX function was actually invoked.
- **ulArgc.** This is the number of parameters passed to the REXX function. It is similar to the **argc** parameter to the C **main()** function.
- **prxArgv.** This is an array of string parameters. It is similar to the **argv** parameter to the C **main()** function. However, instead of an array of **char ***, **prxArgv** is an array of **RXSTRING** structures. Each **RXSTRING** contains a parameter and its length.
- **pszQueueName.** This is the name of the currently defined REXX external data queue.
- **prxRet.** This is a pointer to an **RXSTRING** structure that the C function fills to return a value from the function to the REXX program.

THE RXSTRING STRUCTURE

In the REXX language, the only data type is the string. Parameters passed to external functions are strings, and the value returned from an external

function is a string. REXX strings are represented in C with the `RXSTRING` structure. It is defined as:

```
typedef struct
{
    ULONG strlength;
    PCH strptr;
} RXSTRING;
```

In C, strings always have a hex 00 character that indicate the end of the string. In REXX, all characters are valid in a string—even the hex 00 character. Therefore, the length of the string needs to be encoded outside of the string data. In the `RXSTRING` structure, the `strlength` field contains the length. The `strptr` field is a pointer to the string, like a `char *` variable in C.

When REXX passes arguments to external functions, it adds an extra hex 00 to the end of each argument. This makes it easy to use the `strptr` fields of the `RXSTRING` in C functions such as `strtod()` and `strtok()`. This feature was added in OS/2 2.0.

The REXX language allows empty (zero length) strings. The C equivalent is a `char *` variable that points to a hex 00 character. When a zero-length string is passed as an argument to a REXX external function, the `strlength` field is set to zero, but the `strptr` field is set to a nonzero value. It is also possible to omit parameters from a function call, as in:

```
FunctionName("a",,"b",,"b")
```

In this example, the second and fourth parameters are omitted. These parameters are passed to REXX, with both the `strptr` and `strlength` fields set to zero.

The C language include file from the IBM Developers Toolkit for OS/2 2.0, `REXXSAA.H`, contains a number of useful macros for implementing external functions. One such macro is `RXVALIDSTRING()`. In the sample code contained in Figure 1, this macro is used when checking parameters. This particular macro returns 1 if the `strptr` and `strlength` fields of an `RXSTRING` structure are both nonzero; otherwise 0 is returned.

```
=====
rxprior.mak
=====

#-----
# Makefile for rxprior.dll
#-----

COPTS = /C+ /W3 /Q+ /Gd- /Ge- /Gm+ /Ti+ /O-
LOPTS = /ST:20000 /NOLOGO /DEBUG

OBSJ = rxprior.obj
LIBS = os2386 rexx

.c.obj:
    icc $(COPTS) /Fo$@ $<

rxprior.dll : $(OBSJ)
    link386 $(LOPTS) $(OBSJ),$@,nul,$(LIBS),rxprior.def;

=====
rxprior.def
=====

LIBRARY RXPRIOR INITINSTANCE TERMINSTANCE
DATA MULTIPLE NONSHARED
EXPORTS
    SysSetPriority
    SysGetPriority

=====
rxprior.cmd
=====

/*-----
 * rxprior.cmd : test the external functions in rxprior.dll
 *-----*/

"@echo off"

/*-----
 * load functions
 *-----*/
if RxFuncQuery("SysSetPriority") then
do
    rc = RxFuncAdd("SysSetPriority","RxPrior","SysSetPriority")
    rc = RxFuncAdd("SysGetPriority","RxPrior","SysGetPriority")
end

/*-----
 * print the current priority
 *-----*/
say "current priority:" SysGetPriority()

/*-----
```

Figure 1. External function sample (continued on page 75)



```
* change the priority a few times, and print info each time
*-----*/

call Test "-1", "NOCHANGE"      , "PROCESS"
call Test "-1", "REGULAR"       , "THREAD"
call Test "-1", "TIMECRITICAL"  , ""
call Test "-1", "FOREGROUNDSEVER", "GARBAGE"

call Test "0"
call Test "0", "NOCHANGE"
call Test "1", "NOCHANGE"
call Test "2", "NOCHANGE"
call Test "3", "NOCHANGE"

call Test "0", "REGULAR"

exit

/*-----
* run one test
*-----*/
Test: procedure expose pid
    delta = arg(1)
    class = arg(2)
    scope = arg(3)

/*-----
* call with different # of parameters to test defaulting
*-----*/
if      (arg() = 1) then rc = SysSetPriority(delta)
else if (arg() = 2) then rc = SysSetPriority(delta,class)
else if (arg() = 3) then rc = SysSetPriority(delta,class,scope)

if (rc <> 0) then
    say "rc =" rc "from SysSetPriority("delta","class","scope")"

/*-----
* get the priority
*-----*/
priority = SysGetPriority()

say "priority =" priority,
    "from SysSetPriority("delta","class","scope")"

return

=====
rxprior.c
=====

/*-----
* rxprior.c : external rexx function to set priority
```

PROCESSING THE EXTERNAL FUNCTION

The processing that an external function needs to perform can be broken into three steps:

1. Process the parameters passed to the function.
2. Perform whatever processing is required, based on the parameters.
3. Set the return value.

Processing parameters. Because REXX cannot prototype functions, the user can call external functions with any number of parameters. It's up to you to determine whether the parameters that are passed in are correct.

In the Figure 1, the C function that implements SysSetPriority() function uses the number of parameters that are passed in and the values from the RASTRING parameters to perform basic checks and provide default values. The first check,

```
if ((u1Argc < 1) || (u1Argc > 3))
    return 40;
```

causes the function to return 40 if no parameters or more than three parameters are passed to the function. If 40 is returned from an external function, it causes the REXX error, *Incorrect call to routine*, to be generated. This error is fatal to REXX programs; generate it only if absolutely required. The rest of the parameter-checking code in the SysSetPriority() function provides default values for missing or incorrect parameters that are passed in.

Recall that all parameters are passed in as strings. Therefore, the external function must convert the received arguments from string format to the appropriate data type. In the Figure 1, after checking the parameters, the SysSetPriority() function converts the strings passed in (or defaults) to the types of values it needs. It then calls the C function, *DosSetPriority()*, to do the real work.

Performing function processing. Once the parameters have been processed and converted into C values, you need to do whatever processing the external function is supposed to do. Generally, you can do whatever you want to do, within the

Figure 1. External function sample (continued on page 76)

capabilities of C. There are a few things to look out for:

- Be aware of what the requirements are for the functions that you are calling. For instance, if you intend to make calls to Presentation Manager functions, the current process will probably need to run in a PM session.
- Be careful of non-reentrant processing. Although there are few programs today that allow REXX macros to run in multiple threads within one process, this capability does exist. The same concerns for multithreaded programming in C are valid for external functions for REXX written in C.

Setting return value. To return a REXX value from the function, you must set fields in the `prxRet` structure, which is passed to the function. If the value returned is numeric, you must convert the number from the format C uses to a string, using `sprintf()`, for instance. The `strptr` of this particular `RXSTRING` structure points to a 256-character-long buffer. If the string returned from the function fits in the buffer, write the string to the buffer and set the `strlength` field to the length of the string. If the string is longer than 256 characters, allocate a buffer for it, using `DosAllocMem()`.

Note that you cannot use `malloc()` to allocate the buffer. Set the `strptr` field to the address returned by `DosAllocMem()`. REXX will call `DosFreeMem()` to free this memory after it has made a copy of it.

The external function written in C must return an unsigned long value. Remember that returning 40 from an external function causes a REXX error to occur in the program running the function; any nonzero return code from an external function will cause a fatal REXX error to occur. Make sure you return 0 from your external function, unless you specifically want to cause a fatal error.

REGISTERING THE EXTERNAL FUNCTIONS IN THE DLL

Before calling the external functions in the DLL from a REXX program, the REXX program must register them.

```

*-----*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define INCL_BASE
#include <os2.h>

#define INCL_REXXSAA
#include <rexxsaa.h>

/*-----
 * prototypes for rexx functions, to make sure we declared everything
 * correctly
 *-----*/

RexxFunctionHandler SysSetPriority;
RexxFunctionHandler SysGetPriority;

#pragma linkage(SysSetPriority, system)
#pragma linkage(SysGetPriority, system)

/*-----
 * this function is called when user invokes SysGetPriority() in REXX
 *
 * expects no parameters
 *-----*/
ULONG APIENTRY SysGetPriority(
    PCHAR    pszName,
    ULONG    ulArgc,
    PRXSTRING prxArgv,
    PSZ      pszQueueName,
    PRXSTRING prxRet
)
{
    PPIB ppib;
    PTIB ptib;
    APIRET rc;

    /*-----
     * call DosGetInfoBlocks()
     *-----*/
    rc = DosGetInfoBlocks(&ptib,&ppib);

    /*-----
     * put priority in return code
     *-----*/
    sprintf(prxRet->strptr,"%08.8lx",ptib->tib_ptib2->tib2_ulpri);
    prxRet->strlength = strlen(prxRet->strptr);

    return 0;

```

Figure 1. External function sample (continued on page 77)



```
}

/*-----
 * this function is called when user invokes SysSetPriority() in REXX
 *
 * expects the following parameters
 *   1 - number between -31 and 31, the delta of the priority to set
 *   2 - optional class: "NOCHANGE", "IDLETIME", "REGULAR",
 *                     "TIMECRITICAL", "FOREGROUNDSEVER"
 *   3 - option scope: "PROCESS", "PROCESSTREE", "THREAD"
 *-----*/
ULONG APIENTRY SysSetPriority(
    PCHAR      pszName,
    ULONG      ulArgc,
    PRXSTRING   prxArgv,
    PSZ        pszQueueName,
    PRXSTRING   prxRet
)
{
    APIRET      rc;
    PSZ         pszDelta;
    PSZ         pszClass;
    PSZ         pszScope;
    ULONG       ulScope;
    ULONG       ulClass;
    LONG        lDelta;

    /*-----
     * make sure there are 1, 2 or 3 parameters
     *-----*/
    if ((ulArgc < 1) || (ulArgc > 3))
        return 40;

    /*-----
     * apply default parameters
     *-----*/
    if (RXVALIDSTRING(prxArgv^0))
        pszDelta = prxArgv^0 .strptr;
    else
        pszDelta = "0";

    if ((ulArgc >= 2) && RXVALIDSTRING(prxArgv^1))
        pszClass = prxArgv^1 .strptr;
    else
        pszClass = "NOCHANGE";

    if ((ulArgc >= 3) && RXVALIDSTRING(prxArgv^2))
        pszScope = prxArgv^2 .strptr;
    else
        pszScope = "PROCESS";
}
```

Figure 1. External function sample (continued on page 78)

To do this, use the `RxFuncAdd()` function. The `RxPrior.cmd` file provides an example of registering the two external functions given in `RxPrior.dll`.

Registering a function associates the function name used in REXX programs with a DLL and an entry point in the DLL (the name of the C function). You can use the same entry point for more than one REXX function. This is useful if several external functions perform similar processing. By examining its first parameter, the external function can determine which function was actually invoked.

Once an external function is registered, it is available to any REXX program in the system. Thus, the person using the external functions only needs to register them once, for example, in `STARTUP.CMD`. A REXX program can check if an external function has already been registered with the `RxFuncQuery()` function.

External functions can also be registered in C. This can be useful with a DLL that has a large number of functions. Instead of requiring the user of the external-function package to call `RxFuncAdd()` for each function, you can provide one external function that loads all the other functions. External functions can be registered from C with the function `RexxRegisterFunctionDll()`.

The `REXXUTIL.DLL` provided with OS/2 uses this technique to register its functions. First, you register the function `SysLoadFuncs` in REXX with the `RxFuncAdd()` function, then you call it. This function calls `RexxRegisterFunctionDll()` on the rest of the functions. `REXXUTIL.DLL` also provides a function, `SysDropFuncs`, to unregister the functions. Unregistering a function is a system-wide operation, much like registration. After unregistering a function, the function will be unavailable to REXX programs running on the system, until it is registered again.

COMPILING AND LINKING THE EXTERNAL FUNCTIONS

When writing external REXX functions in C, include the following lines for each external function you write:


```
RexxFunctionHandler FunctionName;
#pragma linkage(FunctionName, system)
```

The first statement prototypes the function. If you use the incorrect argument types or forget an argument, this prototype will cause a compiler error to be generated. An external function that does not have the correct prototype can cause the external function to behave incorrectly or cause an access violation.

The `#pragma` statement ensures that the function has the system linkage (as opposed to the `optlink` linkage) for C Set/2.

A make file is provided with the sample in Figure 1. The options used will compile and link the source with C Set/2 compiler and the `link386` program.

A `.def` file is also included. Each of the external functions in C needs to be exported from the DLL.

The DLL must be linked with `REXX.LIB`, in addition to the standard C libraries and any additional libraries that are needed.

DEBUGGING THE EXTERNAL FUNCTIONS

External functions contained in a DLL can be debugged with the IPMD debugger. Invoke IPMD with the command:

```
IPMD cmd /c myprog
```

where `myprog.cmd` is a REXX `.CMD` file containing calls to the external functions. Once IPMD starts, set a load breakpoint on your DLL and have IPMD start the process. Your DLL will load when the first function is called in it. At this point, set breakpoints, single step, and so on, within the debugger.

If you need to recompile your DLL after using it, the link step may fail with an error indicating that the run file (the DLL) cannot be opened. This is because when a REXX program is run from a `CMD.EXE` command prompt, it is run in the current OS/2 process. Specifically, your current `CMD.EXE` session has the DLL loaded. In fact, all of the currently-open `CMD.EXE` sessions might have the DLL loaded. OS/2 does not allow DLLs

```
/*-----
 * convert delta to a number
 *-----*/
lDelta = strtol(pszDelta,NULL,10);

/*-----
 * convert class to a pre-defined constant
 *-----*/
if (!strcmp(pszClass, "NOCHANGE"))
    ulClass = PRTYC_NOCHANGE;

else if (!strcmp(pszClass, "IDLETIME"))
    ulClass = PRTYC_IDLETIME;

else if (!strcmp(pszClass, "REGULAR"))
    ulClass = PRTYC_REGULAR;

else if (!strcmp(pszClass, "TIMECRITICAL"))
    ulClass = PRTYC_TIMECRITICAL;

else if (!strcmp(pszClass, "FOREGROUNDSEVER"))
    ulClass = PRTYC_FOREGROUNDSEVER;

else
    ulClass = PRTYC_NOCHANGE;

/*-----
 * convert scope to a pre-defined constant
 *-----*/
if (!strcmp(pszScope, "PROCESS"))
    ulScope = PRTYS_PROCESS;

else if (!strcmp(pszScope, "PROCESSTREE"))
    ulScope = PRTYS_PROCESSTREE;

else if (!strcmp(pszScope, "THREAD"))
    ulScope = PRTYS_THREAD;

else
    ulScope = PRTYS_PROCESS;

/*-----
 * call the DosSetPriority() function
 *-----*/
rc = DosSetPriority(ulScope,ulClass,lDelta,0);

/*-----
 * set return value
 *-----*/
sprintf(prxRet->strptr,"%ld",rc);
prxRet->strlength = strlen(prxRet->strptr);

return 0;
}
```

Figure 1. External function sample (continued from page 77)

that are currently loaded to be erased or written to. To remove the DLL, you need to unregister each of the functions. External functions can be unregistered with the `RxFuncDrop()` function in REXX or the `RexxDeregisterFunction()` function in C. After unregistering the functions, you must exit all, currently-open, CMD.EXE sessions. OS/2 will then drop the DLL from the system. At this point, you can start new CMD.EXE sessions and erase or relink the DLL.

REXX MACROS IN THE CURRENT PROCESS

REXX macros have a significant difference from other OS/2 executable files: they do not start a new process. A REXX .CMD file, run from a CMD.EXE session, runs in the same process as the CMD.EXE program. OS/2 does not start a new process for the program or end it when the program terminates.

This is important, because OS/2 resources are not cleaned up when a REXX macro ends. For example, memory allocated with `malloc()` or `DosAllocMem()` from an external function is not freed when the REXX program that used the external function ends. If you need to use these functions to allocate memory from within an external function, free the memory before the external function returns or it will remain allocated within the CMD.EXE process. If you forget to free the memory, you won't be able to free it later, because you will have lost the pointer value. The only way to free the memory will be to exit the current CMD.EXE process.

This also has implications for `DosExitList()` processing. If your external

function uses another DLL that depends on `DosExitList()` processing, this processing won't occur until after the CMD.EXE process has terminated.

You can get around this limitation by writing a C program that invokes REXX macros with the `RexxStart()` function. The program can register the external functions before starting the REXX macro with the `RexxRegisterFunctionDll()` or `RexxRegisterFunctionExe()` functions. The first argument to the program should be used as a REXX macro name, and the remainder should be concatenated to form the parameter string to be passed to the REXX macro.

This C program can then be used with the `EXTPROC` facility of OS/2. In a .CMD file, make the first line:

`EXTPROC myCprog`

where `myCprog` is the name of the C program.

When OS/2 runs the .CMD file, it recognizes the `EXTPROC` statement and starts the `myCprog` program with the parameters specified on the command line for the .CMD file. The C program is then in control. A new process starts and will terminate when the program ends. Therefore, OS/2 resources, such as memory and file handles, will be cleaned up when the `myCprog` program ends.

You should unregister the REXX functions that were registered by the C program, before the C program exits.

EXTERNAL FUNCTION SAMPLE

The sample code implements two functions: `SysGetPriority()` and `SysSetPriority()`. These functions are used to query and set the priority of the current process.

SysGetPriority(). This function expects no parameters. It returns an eight-digit hex string that indicates the priority of the current process. The function returns the priority from a field in the info-blocks structure returned by the `DosGetInfoBlocks()` function.

SysSetPriority(delta,class,scope). This function expects the `delta` parameter to be a number from -31 to 31; the `class` parameter to be one of the strings, `NOCHANGE`, `IDLETIME`, `REGULAR`, `TIME-CRITICAL`, or `FOREGROUNDSEVER`; and the `scope` parameter to be one of the strings, `PROCESS`, `PROCESSTREE`, or `THREAD`. The parameters correspond to the values expected by the `DosSetPriority()` function. The external function calls this function and returns the return code from that function. The `delta` parameter must be passed in, but the other two are optional. Missing or incorrect parameters cause default values to be used.

`RxPrior.cmd` tests the external functions by calling `SysSetPriority()` a number of times with different parameters. After each call to `SysSetPriority()`, `SysGetPriority()` is called to determine what the priority was set to. The resulting value is printed as output from the program.

CONCLUSION

The REXX language allows for expansion of its function base through external functions. Based on that design, and with the help of these tips and explanations, you are capable of extending the language to meet your needs. Go for it!

Andrei Malacinski, IBM Programming Systems, Cary N.C., is a member of the Distributed Application/2 programming team. He joined IBM in 1990, after receiving degrees in mathematics and computer science from Indiana University.

Patrick Mueller, IBM Programming Systems, Cary N.C., is a member of the Object Oriented Strategy Department. He joined IBM in 1985, after receiving degrees in mathematics and computer science from Purdue University. He can be reached at pmueller@vnet.ibm.com.

REFERENCES

OS/2 2.0 Technical Library, Procedures Language 2/REXX User's Guide, (IBM Doc. S10G-6269).

OS/2 2.0 Technical Library, Procedures Language 2/REXX Reference, (IBM Doc. S10G-6268).



Open Client / Server Solutions Conference

May 31 to June 3, 1994
Professional and Technical Education Department
IBM International Education Centre



**Is your enterprise ready
for the next decade of business**

?

*Are you able to plan and to implement a **Rightsized**,
well **Architected, Harmonised**, and
Heterogeneous
Information Technology environment*

?

*Do you know how **Open IBM** is*

?

If you hesitated in your answer...

YOU MUST NOT MISS THIS EVENT !!!

It will provide your Technical Workforce
with the skills your enterprise requires:

For further information you may contact
your nearest IBM branch office or directly:

Mr. Jose Maria Fernandez,
Conference Manager in Belgium
Phone: 32-2655-5799. Fax: 32-2655-5739
Mail: BEIBM4P at IBMAIL
Internet: JFERNANDEZ at vnet.ibm.com

ADVERTISER INDEX

Reader Service	Company Name	Page
65	Borland International	COV4
57	Burton Systems Software	86
22	Creative Systems Programming	41
59	Development Technologies, Inc	87
5	Digitalk.	7
28	dSoft Development	47
31	Globalink, Inc.	51
20	Gpf Systems Inc.	35
15	Graphical Software Interfaces	25
60	Hardy Software Systems	87
7	HockWare	13
19	IBM Canada	33
24	IBM Canada	45
10	IBM GIK	10
54	IBM International Education Center	80
	IBM Personal Software Products	9
	IBM Personal Software Products	11
35	IBM Software Vendor Operations	57
52	IBM Technical Interchange	71
16	Ingres	27
14	Impact Software	25
36	Indelible Blue, Inc.	57
46	Intersolv	63
9	Kaseworks	17
27	Levine Computer Consultants	47
17	Mathematica Journal	29
21	MHR Software & Consulting.	40
23	Microformatic	39
29	Mitnor Software	48
2	Micro Focus	2
51	Object Design	70
37	Op-Tech Data Processing.	57
63	OS2 Applications Directory	91
45	OS2 Magazine	61
26	Perez Computing Service.	46
3	Programmer's Paradise.	4
13	Quadron	25
32	Rhetorex Inc.	52
8	Rimstar Technology.	15
47	SE International	64
61	ScheduPerformance	89
64	Sequiter Software	COV3
25	Softbridge, Inc	46
30	SofTouch Systems	49
48	Software Development	65
56	Special Reports.	85
11	Stirling Group	23
58	Taylormade Systems	86
55	Van Nostrand Reinhold.	81
38	Walnut Creek CDRom	57
1	Watcom C for IBM PC's	COV2-1

VNR KNOWS OS/2®

CLIENT/SERVER PROGRAMMING

WITH OS/2 2.1, Third Edition by Robert

Orfali and Dan Harkey. OS/2 MONTHLY called the 2nd Edition *"The OS/2 Book of the Year!"*

Others said individual chapters were by themselves *"worth the price of the book."*

This new edition keeps you state-of-the-art in OS/2 client/server developments. \$39.95 0-442-01833-9

OS/2 2.1 CORPORATE PROGRAMMER'S HANDBOOK

by Nora Scholin, Martin Sullivan and Robin Scragg. A quick and easy modular approach to writing software programs. Gives code examples for using OS/2 2.1 to its full potential.

\$39.95 0-442-01598-4



OS/2 2.1 REXX HANDBOOK: Basics, Applications and Tips

by Hallett German. A one-stop guide to one of the most popular command languages. Combines a basic tutorial with fast answers to all REXX questions.

\$29.95 0-442-01734-0

WRITING OS/2 2.1 DEVICE DRIVERS IN C, Second Edition

by Steven J. Mastrianni. Defines each type of driver and how to fit them into your system with tips on debugging, tuning, and enhancing driver performance. Includes disk with source code for four complete drivers plus all source code found in book.

\$39.95 with disk. 0-442-01729-4

Van Nostrand Reinhold

115 Fifth Avenue New York, New York 10003



To place an order, or for a complete listing of VNR titles, call 1-800-544-0550 (Dept. Z 1555) or GO VNR on the CompuServe® Network.

OS/2® is a registered trademark of IBM Corporation.

Circle Reader Service Number 55





Buyers' Guide

The staff of OS/2 Developer put together this special section to guide you through the various database development tools available for OS/2. The products described in this guide are based on surveys sent to vendors; they are provided as a service to you and are free to vendors. These listings do not represent an endorsement by OS/2 Developer. Although every effort has been made to ensure accuracy, we do not assume any responsibility for error or omission in this section. Please contact the companies directly for more information.

Database Development Tools Buyers' Guide

THE ASK GROUP INC.

Circle No. 101

ASK INGRES 6.4 Intelligent Database is a member of the Database and Connectivity Product Unit's family of intelligent relational database management products and services. It helps companies manage not only data, but knowledge (business rules) and objects (non-conventional data such as maps and pictures) as well. ASK INGRES operates on many hardware platforms and enables customers to develop and operate information systems for the client/server environment. Call for pricing information.

The Ask Group Inc., 1080 Marina Village Pkwy., Alameda, Calif. 94501, (800) 446-4737 or (510) 769-1400, fax (510) 748-2670.

BACHMAN INFORMATION SYSTEMS INC.

Circle No. 102

Bachman/DLL Generator for DBM and DB2/2 is a part of Bachman's family of 17 products for database access, design, modeling, and analysis. It provides a bi-directional interface between the OS/2 databases, Extended Edition Database Manager and DB2/2, and Bachman/DBA, a graphical database design environment. Bachman/DDDL Generator for DBM and DB2/2 allows users of Bachman modeling and database design products to implement databases in the Database Manager or DB2/2 relational databases. It also allows database administrators to re-engineer existing databases from another DBMS into DBM or DB2/2, or from DBM or DB2/2 into another DBMS. Price: \$3,500; quantity discounts apply.

Bachman Information Systems Inc., 8 New England Executive Pk., Burlington, Mass. 01803, (617) 273-9003, fax (617) 229-9904.

BORLAND INTERNATIONAL

Circle No. 103

Borland C++ for OS/2 1.0 is a development system for creating OS/2.x applications. The compiler produces a 32-bit object code and supports development of multithreaded applications, mixed-mode programming in OS/2, and the OS/2 calling conventions. Features, such as on-line IPF help and documentation, PM-hosted environment, integrated GUI debugger and the new resource Workshop for OS/2, aid in designing and modifying OS/2 resources like bitmaps, icons, dialogues, strings, and menus. Price: SRP, \$495; Borland C++ Customers, \$149.95.

Borland International, 100 Borland W., Scotts Valley, Calif. 95066-3249, (800) 331-0877 or (408) 431-1000, fax (408) 431-9119.

COMPUTER ASSOCIATES INTERNATIONAL INC.

Circle No. 104

CA-Realizer 2.0 for OS/2 is a visual programming environment, handles the mechanics of event-driven programming, message passing, and process sharing. It also handles GUI programming, including memory management, event handling, and multitasking. It combines visual development tools, Programmable Application Tools, and a structured superset of BASIC extended to access Windows and OS/2 objects and resources. Other enhancements include FormDev, which allows developers to design visual elements of a program by using



standard graphical objects, true record structures, nested families, matrix functions, and multidimensional arrays in up to 30 dynamically expandable dimensions. CA-Realizer includes built-in tools such as charts (pie, candlestick, I-beam, horizontal-bar, filled-area, and Grant types), spreadsheets, boards, logs, animation, graphics tablets, and a scheduler. It supports the Open Database Connectivity standard (ODBC) and imports and exports Xbase (.DBF) files, generating data entry forms in FormDev. Applications built with CA-Realizer may be distributed free of royalty or license fees. Price: \$99, until March 31, 1994; \$247, after March 31, 1994.

Computer Associates International Inc., One Computer Associates Plaza, New York, NY 11788-7000, (800) 225-5224 or (516) 342-5224, fax (516) 342-5329.

CONSYST SQL INC.

Circle No. 105

SQL Design 2.0 is a fourth generation productivity tool built as a shell on many SQL databases. It is used for development and maintenance of applications. Its centralized repository concept is the same used in computer aided software engineering tools where all data and application elements are stored. This repository is linked to the generated applications and eliminates data and function duplication. Price: \$750-5,000.

Consynt SQL Inc., 15 Mont-Royal Street Ouest, Suite 120, Montreal, Quebec, Canada, H2T2R9, (514) 849-7431, fax (514) 849-8125.

COPIA INTERNATIONAL LTD.

Circle No. 106

AccSys for Paradox 4.0 provides C, QuickBasic, and/or Visual Basic programmers with access to the database routines found in Paradox, as well as the ability to create, read, write, modify, and update Paradox files without having to learn the internal file formats. Paradox's interface provides users with tools for control over Paradox primary and secondary index files, as well as tables. AccSys supports Borland C, C++, Microsoft C, Zortech, WATCOM. It allows for standalone program construction. Network-compatible. No runtime royalties. Price: \$1195, for product with source.

AccSys for dBASE 2.92 enables C and/or Visual-Basic programmers to update xBASE, dBASE III and IV file types. It works with MDX and NDX index files and DBT memo files and also handles

the DBF, NDX, and DBT files in both dBASE III (PLUS) and IV formats. AccSys supports Borland C, C++, Microsoft C, Zortech, WATCOM, FoxPro, and Clipper. It allows for standalone program construction. Network-compatible. No runtime royalties. Price: \$495.

Copia International Ltd., 1342 Avalon Ct., Wheaton, Ill. 60187, (708) 682-8898, fax (708) 665-9841.

DATA ACCESS CORPORATION

Circle No. 107

DataFlex 3.01, an application development environment, combines an object-oriented fourth generation language, a WYSIWYG application generator, a relational database management system, and a set of built-in productivity tools and utilities. DataFlex is portable and is available in several languages, including Danish, English, French, German, Italian, Norwegian, Portuguese, and Swedish. Price: \$795 (single user); \$1450 (multiuser).

Data Access Corporation, 14000 S.W. 119 Ave., Miami, Fla. 33186-6017, (800) 451-FLEX or (305) 238-0012, fax (305) 238-0017.

DESKTOP AI

Circle No. 108

X2c 1.15 Portable Xbase Compiler for OS/2 is an open Xbase compiler used to create OS/2 32-bit compiled .EXEs. It creates a C version of the application by using a native C compiler; for 32-bit applications, X2c uses the Borland C++ compiler. X2c supports characteristics of Xbase variables, including macroed expressions with UDF calls and type re-assignment; features of dBASE III+, Clipper (S'87), Foxbase+ 2.1; and many features of FoxPro, such as user defined functions, VALID (), @/prompt /menu, 1&2 dimension arrays, and array functions: DBEDIT (), ACHOICE (), MEMOEDIT (), READ MENU, windows, buttons, BROWSE. All support functions are available in C source, and C routines can be added. Price: \$1,000 (call for special prices).

Desktop Ai, Box 59, Georgetown, Conn. 06829, (203) 255-3400, fax (203) 259-8853, e-mail sales@dtop.com or CIS: 76137,3727.

DIGITALK

Circle No. 109

PARTS Relational Database Interface 2.0, a companion product to PARTS Workbench, allows the user to create graphical client/server applications that use SQL databases. ANSI standard SQL and SQL



extensions are supported for each database. The product supports the following databases: Microsoft/Sybase SQL Server, Oracle, IBM DB2/2, IBM DB2 through the MicroDecisionWare gateway, IBM DB2, OS/400 and SQL/DS through the IBM DDCS/2 gateway, IBM Extended Services Database Manager, Gupta SQLBase, dBASE III Plus, dBASE IV, Novell NetWare SQL Databases, XDB Database Management System, Btrieve, Excel files, and text files. The PARTS relational database interface allows the user to create graphical front-end applications that work independently of the database they access, making it possible to create client/server applications that are portable across different databases. Price: \$995.

Digitalk, 5 Hutton Centre Dr., Santa Ana, Calif. 92707, (800) 922-8255 or (714) 513-3000, fax (714) 513-3100, e-mail sales@digitalk.com.

DSOFT DEVELOPMENT INC *Circle No. 110*
dbfLIB 1.06 Programmer's Library is a C/C++ programmer's library for accessing dBASE data files and indexes. It includes royalty-free, dynamic-link libraries for Windows, OS/2 1.3 (16-bit), and OS/2 2.x (32-bit) and supports Microsoft, Borland, and IBM Compilers. dbfLIB provides a consistent set of APIs across DOS, Windows, and OS/2 platforms, allowing the user to develop new applications than can share dBASE files with dBASE programs currently in use, as well as other programs that use dBASE files. Price: \$159.

dSOFT Development Inc., 4710 Innsbruck Dr., Houston, Tex. 77066, (713) 537-0318, fax (405) 360-3045, CompuServe 70562,1044.

EASEL CORPORATION *Circle No. 111*
ENFIN 2.9 is an object-oriented, visual development environment used for building a variety of client/server applications. It is based on the Smalltalk programming language. ENFIN supports reusability and portability. Applications built with ENFIN are portable between OS/2, Windows, and UNIX. Price: \$5,995.

Easel Corporation, 25 Corporate Dr., Burlington, Mass. 01803, (800) OBJECTS

(625-3287) or (617) 221-2100, fax (617) 221-6899.

FAIRCOM USA/FAIRCOM EUROPE *Circle No. 112*
c-tree Plus File Handler 6.0 rel.e allows users to use low-level routines or take advantage of the ISAM routines for high speed random or sequential access. It has C source code and is ported to 100+ environments. It is designed for single-user, multiuser or client-side application development. c-tree Plus File Handler includes: transaction processing, fixed-variable length records and keys, file level alternate collating sequence, duplicate keys and/or automatic sequence numbers, native operating system I/O utilization for portability, dynamic space reclamation, high speed hashed data/index caching, multiple simultaneous sets/batches, and resource records superfiles. It has ISAM functionality and is ANSI prototyped. Royalty-free. Call for a complete list of features. Price: \$595.

r-tree Report Generator 1.2e provides multiline reports by handling aspects of report generation. The user calls the r-tree report function, which then reads c-tree data files, performs calculations, monitors control breaks and accumulators, and produces a formatted report. r-tree comes with C source code and supports many C compilers. It requires c-tree Plus. Specify platform: DOS (Windows 3.x, OS/2, UNIX TAR, QNX 4.X, Coherent, and Xenix) or Macintosh. Specify 5.25-inch or 3.5-inch disks. Limited 30 day guarantee. Price: \$295.

FairCom SQL Server 6.03.16 is a database server that utilizes the c-tree Plus Application Programmer's Interface (API) plus an ANSI 1986 level 2 SQL engine. It provides data integrity through multiuser transaction processing and includes client side source code. Specify operating environment: DOS, OS/2, AT&T UNIX, SCO UNIX, SCO Xenix, Apple A/UX, Interactive UNIX, QNX, Banyan VINES, Sun SPARC, HP9000, RS/6000, 88OPEN. Specify 5.25-inch or 3.5-inch disks. Limited 30 day guarantee. Price: \$445-3,795,

depending upon platform; number of users. Call FairCom.

FairCom Server 6.03.16 is a transaction processing server. Its features include: industrial quality transaction processing, including full commit and rollback; intermediate save points and complete logging; automatic log management; restart/disaster recovery; user passwords; access security and on-line administration; deadlock detection/resolution; and read/write locks at the record/individual key level. Specify operating environment: DOS Extended/Windows Hosted, OS/2, AT&T UNIX, SCO UNIX, Interactive UNIX, QNX, SUN SPARC.HP 9000, RS/6000, 88OPEN, Banyan VINES, Data General Avion, NLM, SCO Xenix, Apple A/UX. Specify 5.25-inch or 3.5-inch disks. Limited 30 day guarantee. Price: \$295-2,495, depending upon platform, number of users. Call FairCom.

Natural Query enables the user to query c-tree Plus databases without programming. The user can formulate queries using English, ANSI standard Structured Query Language or Query-By-Example. In addition, sessions maintained for reuse of queries/reports. Natural Query also allows the user to customize personal vocabulary. The Developer's Package allows the user to add NQ's capabilities to developer applications, giving end users the ability to query in English/SQL/QBE. Specify DOS or OS/2. Specify 5.25-inch or 3.5-inch disks and whether c-tree/c-tree Plus. Limited 30 day guarantee. 2.5 lbs. Price: \$169, end user; \$395, developer's package.

d-tree Development Tool 3.2 rel.e offers beginning programmers a bridge to the control, flexibility, and portability of C development. Its features include productive front end for c-tree; complete portable screen handler; menu/HELP management; data dictionary; file/field level editing/validation; and resource swapping; runtime control of program resources. Specify 5.25-inch or 3.5-inch disks (includes DOS & UNIX TAR). Version 3.2 release e. Limited 30 day guarantee. Price: \$495.

FairCom USA, 4006 W. Broadway, Columbia, MO 65203, (800) 234-8180 or

DON'T MISS OUT!

Now you can get Special Reports for *Virtual Reality*, *Neural Network*, and *AI in Finance* at incredibly **low** prices! This offer won't last forever, so order **now**. Don't take a chance! **Don't miss this incredible offer!**



☒ **YES!** Please send me the Special Reports that I have marked below:

of
copies

- _____ 1992 Virtual Reality Special Report \$9.95
_____ March 1993 Virtual Reality Special Report \$9.95
_____ August 1993 Virtual Reality Special Report \$9.95
_____ 1992 Neural Network Special Report \$7.95
_____ 1993 Neural Network Special Report \$7.95
_____ Neural Net Primer Special Report \$7.95
(Available in December 1993)
_____ AI in Finance Special Report \$7.95

Name _____

Company _____

Address _____

City _____

State _____ Zip _____

Prepayment is Required.

Subtotal for _____ copies of all VRSR @ \$9.95 each

Subtotal for _____ copies of all NNSR @ \$7.95 each

Subtotal for _____ copies of AIFSR @ \$7.95 each.

TOTAL

FAX (415) 905-2233 or call (800) 444-4881

Allow up to 6 weeks for delivery of your first issue. Payable in U.S. dollars. Please make checks payable to Miller Freeman. Canadian GST #124513185

Circle Reader Service Number 56



(314) 445-6833, fax (314) 445-9698. Fair-Com Europe, Via Sottocorna 15/17, 24021 Albino (BG) Italy, 035/773464, fax 035/773806.

GPF SYSTEMS INC. Circle No. 113

Gpf 2.0 Single Platform is a WYSIWYG CUA 91 application development/source code generator for OS/2 PM 2.x. Its client-window support permits the user to manipulate controls including size, position, font, and color. In addition, Gpf allows the user to link built-in and user function, IPF help panels, and help messages at design time without leaving the graphics interface. It uses SQL code generation for OS/2 Database to fill list boxes from existing tables. Gpf supports multithread designs and generates ANSI "C" code as well as ancillary files. Price: \$495.

Gpf Systems Inc., 30 Falls Rd., P.O. Box 414, Moodus, Conn. 06469, (800) 831-0017 or (203) 873-3300, fax (203) 873-3302.

GSI (GRAPHICAL SOFTWARE INTERFACES) INC.

Circle No. 114

SQL Objects ++ 1.1 Database Class Library is a collection of optimized C++ database classes that provide direct access to SQL and non-SQL databases. It provides access for each database the user works with. The user is not restricted to a handful of common C APIs and can access DBMS-specific calls with powerful C++ classes. It provides an object-oriented approach to database access. Each powerful class represents a specific DBMS that is derived from a common base class. The user can derive new classes with extensions. SQL Objects++ Database Class Library has 32-bit database access and multiplatform support. It enables the user to build one application that accesses DB2/2, Oracle, Sybase, SQL Server, Netware SQL, Btrieve and others that can also run in the OS/2, UNIX, DOS, Windows, and NT environments. No royalties. The OS/2 version also sup-

ports SOM (system object model). Price: \$699.

GSI (Graphical Software Interfaces) Inc., 47 Stonewall St., Cartersville, Ga. 30120, (800) 876-6585 or (404) 382-6585, fax (404) 382-6374.

GUPTA CORPORATION Circle No. 115

SQLBase Server 5.1 aids the user in creating graphical applications, OLTP (on-line transaction processing) and decision support. Its features include client/server architecture crash recovery, password protection, on-line backup, remote monitoring and diagnostic tool. Multiple users can simultaneously perform data entry, browse between PC front end application and multiple sets of data records, and do background reporting at anytime while data is being updated. Front ends for SQLBase Server include: SQLWindows, Quest, SQR, Access SQL, C, COBOL, Clipper, dBase/SQL, Forest & Trees, JAM,

TLIB™ Version Control For DOS, OS/2 and Windows-NT

• **The experts loved TLIB 4:**

"...amazingly fast... TLIB is a great system." **PC Tech Journal**
"TLIB has features and power to spare... TLIB is easy to use and the fastest of the reviewed packages." **Computer Language**
"I will not program without it." **Uptime Magazine**

• **Now TLIB 5.0 adds:**

Automatic branching. Automatic version labeling across branches. User defined **promote** structures, for staged development. Exclusive whole-level **change migration** for customized software. N-way-tree version numbers. Branch and full locking. OS/2 & NT support.

• **Plus the features they loved in TLIB 4:**

Check-in/out locking. Branching, for parallel development. Keywords. Full binary file support (does not depend upon CRs in the file like other products). Wildcard and list-of-file support; can create lists by scanning source code for includes. Can merge (reconcile) multiple simultaneous changes and undo intermediate revisions. Network and WORM optical disk support. Mainframe-compatible delta generator for Pansophic, ADR, IBM, Sperry formats. Includes integrated PD MAKE by L. Dyer; also integrates with Opus™ MAKE, Slick™ MAKE, others.

MS-DOS \$139, OS/2 & NT (with MS-DOS) \$195 + shipping.
5 station network: MS-DOS \$419, OS/2 \$595. Call for other sizes.

 **Burton Systems Software**
PO Box 4156, Cary, NC 27519 (919) 233-8128
FAX: 233-0716

Print Utility/2 for OS/2



Save Paper With File Lister/2

- ▶ Reduce the amount of paper you use by printing text files to 1,2, or 4 pages per sheet of paper.
- ▶ If your printer supports duplex printing, get up to 8 pages per sheet of paper.
- ▶ Take advantage of the installed system fonts.
- ▶ Great for compiler listings.

Use File Lister/2 As A File Viewer

- ▶ View binary files such as INI files in hexadecimal dump format.

Get A Screen Capture Utility

- ▶ Capture Presentation Manager Windows or the OS/2 Desktop to a file, printer, or the system clipboard.
- ▶ Paste images into a word processor and create great looking User Manuals.



(205) 344-0672



Taylor Made Systems
Mobile, AL 36618

Objectview, PC NOMAD, Quicksilver/SQL, R&R Reportwriter, SQLVision (for Lotus 1-2-3 and Microsoft Excel), SUPERBASE, and VP-Expert. Requirements for OS/2 Client: IBM PC/AT, PS/2 models 30-286, 50, 60, 70, 80; and compatibles; 4MB RAM; 3MB hard drive disk space; and OS/2 1.0 or higher. Requirements for OS/2 Server: IBM PC, XT, AT, PS/2 models 30-386, 50, 60, 70, 80; and compatibles; 4MB RAM (entry level configuration); 10MB hard disk space (entry level configuration); and OS/2 1.0 or higher. SQLBase Server supports the following networks: NetBIOS compatible LANs including, NetWare 386 & Novell 2.11 or higher, IBM LAN Server, 3COM 3+ Open, Banyan VINES, and Microsoft LAN Manager. Price: \$995-9,995.

Gupta Corp., 1060 Marsh Rd., Menlo Park, Calif. 94025, (800) 876-3267 or (415) 321-9500, fax (415) 321-5471.

IBM CORPORATION

Circle No. 116

Database Server for OS/2 (DB2/2) 1.0 is a 32-bit relational database for use in either a client/server LAN configuration or as a single-user desktop database. As a LAN database server, it supports application clients on DOS, DOS Windows, and OS/2. Functionality includes referential integrity, DRDA support, query manager, database administrative tools, and an OS/2 command line processor. Upgrades are available for current users of the Database Manager on OS/2 Extended Edition 1.3 and Extended Services 1.0 migrating to OS/2 2.0.

Price: \$425, single user; \$2,495, server.

Distributed Database Connection Services/2 (DDCS/2) 1.0 provides the DRDA communications between LAN and enterprise data. LAN client applications use the DB2/2 database server to read or write host databases,

including DB/2MVS, SQL/DS, and OS/400's relational database. DB2/2 single-user databases use a single-user version of the DDCS/2 gateway.

Price: \$500, single user; \$4,680, multiple user.

IBM Corp., 895 Don Mills Rd., Don Mills, Ont., Canada, M3C 1W3, (416) 448-2637, fax (416) 448-2114.

IMAGESOFT INC.

Circle No. 117

CommonBase 1.2* is a portable C++ application framework for developing database applications. It hides the intricacies of each database API and removes learning or using SQL from the developer. Applications developed using CommonBase are portable across databases and operating systems. Price: \$1,000.

ImageSoft Inc., 2 Haven Ave., Port Washington, N.Y. 11050, (800) 245-8840 or (516) 767-2233, fax (516) 767-9067, e-mail mcdhup!image!os2.



DeskMan/2™

The Desktop Manager for OS/2®

Now includes a WPS snapshot facility and VUEMan/2™

Install, configure, backup, restore, secure, and otherwise manage Workplace Shell™ objects, locally or remotely, all with access controls.

VUEMan/2, included with **DeskMan/2**, provides a powerful virtual desktop, including security, window management, screen savers, and more.

"I installed OS/2 2.1 golden code, and DeskMan/2 saved me at least 4 hours of tedious labor. I can't imagine being without it."

David Barnes
Senior staff member - IBM's
OS/2 Executive Briefing Center

DeskMan/2 is fully CID enabled, and supported by LAD/2.

DevTech

Development Technologies, Inc.
Software Development & Technology Transfer

308 Springwood Road • Forest Acres, SC 29206-2113
(803) 790-9230



for LAN Server

You'll never use your debugger again!

Footprints™



a native 32-bit OS/2 trace facility

Debugging
Remote support
Execution verification
Performance monitoring
and more ... for only \$99.00

Once you define traces in your code, you can turn them on and off from our **PM** interface at runtime without recompiling. You can use 64 different traces to track thousands of variables and data areas. **For more information, call one of our developers today.**



(713) 871-1448, Fax (713) 871-1449

Hardy Software Systems, Inc.

4801 Woodway, Suite 415W, Houston, Texas 77056

Visa / Mastercard / Discover accepted. Add \$ 5.95 shipping and handling.

Circle Reader Service Number 59

Circle Reader Service Number 60



IMPALA SYSTEMS INC. Circle No. 118

IMPALA DB Advisor 1.0 is a database productivity tool that provides information, such as in-depth reports, on DB2/2 or Extended Services database structures and objects. IMPALA DB Advisor aids in problem solving, analysis and reporting, and applications development testing. Price: Please inquire. IMPALA DB Advisor 1.0 will ship March 31, 1994.

Impala Systems Inc., P.O. Box 2210, 11830 S.W. Kerr Pkwy., Lake Oswego, Oreg. 97035, (503) 452-1953, fax (503) 452-1179.

INFERENCE CORPORATION

Circle No. 119

ART*Enterprise 1.0 for OS/2 is an object-oriented client/server development tool for building computing applications. It features point-and-click generation of GUI, database mapping and application objects, multimedia, multideveloper support, case based retrieval (CBR) of unstructured information, and rules. ART*Enterprise allows creation of applications that link multiple heterogeneous databases and are portable across OS/2, Macintosh, Windows, Windows NT, UNIX, and MVS. Price: \$6,995. Beta on OS/2 available now. General availability: March 15, 1994.

Inference Corp., 550 N. Continental Blvd., El Segundo, Calif. 90245, (800) 322-9923 or (310) 322-0200, fax (310) 322-3242.

INFORMATION BUILDERS Circle No. 120

PM/FOCUS 1.5 includes a graphical toolset with File Painter, Forms Painter, and Query Painter for building applications. Application components are transformed into graphic objects that can be moved, manipulated, and activated with a point, click, and drag of the mouse. PM/FOCUS Query Painter transforms database fields, record selections, and report headings and footers into selectable objects. PM/FOCUS becomes LAN-ready with the purchase of FOCUS/Database Server available in 8, 16, 32, and larger user configura-

tions. It comes with its own database engine and allows the user to port existing FOCUS databases, create new ones, or use optional interfaces to create GUI applications for DB2/2 and many other SQL databases. The price is \$798.

Information Builders, 1250 Broadway, New York, N.Y. 10001, (800) 969-INFO (4636) or (212) 736-4433, fax (212) 695-3247.

INNOVATIVE DATA CONCEPTS

Circle No. 121

TFILE for OS/2 2.12 provides routines for manipulating database files including reading, writing, and creating database files; inserting and removing records; and indexing by one key or multiple keys. It is an indexed file-access system that is dBase III/III+ compatible. In addition, it can also maintain DBF, DBT, and NDX files, as well as Clipper-compatible indexes. TFILE uses dBase III file formats for data, index, and memo files, and has a pair of routines to translate database records from dBase III format to a format usable by TCXL's Data Entry System, and back again. A tool is provided to generate source code for data structures. Price: \$59.

Innovative Data Concepts, 122 North York Rd., Suite 5, Hatboro, Pa. 19040, (800) 926-4551 or (215) 443-9705, fax (215) 443-9753.

INTEGRATED INFONET TECHNOLOGY (INIT)

Circle No. 122

BusinessLink (BL) 1.0 is an object-oriented, 32-bit application and information system (IS) management software package that runs on OS/2 2.1 and utilizes OS/2 Presentation Manager (PM), DB2/2 Database Manager, Communication Manager, and the OS/2 LAN configuration for both single user and multiuser environments. It allows the user to create tables and add field names and values. It can be uploaded, integrated and worked within the existing database system. BusinessLink consists of the InIT supplied databases and the user-defined "customized interface" to the RDBMS that is used to

store data, information query and retrieval. The following modules are available: Request for Quotation, Bid Evaluation, Vendor Selection for award, Proposal Preparation, Distributor Pricing comparison and Vendor Prices determination, Proposal Analyses, Purchase Order issuing, Invoicing and Product & Service Tracking. BusinessLink offers a standalone version, distributed Client/Server system, and an OS/2 LAN Server with DOS/Windows client and OS/2 client accessibility. C++ Programming approach; application design database management system; GUI implementation. Price: \$799.

Integrated InfoNet Technology, 25 Mauchly, Suite 320, Irvine, Calif. 92718, (714) 753-7868, fax (714) 753-7877 or (714) 753-1777.

INTELLIGENT ENVIRONMENTS

Circle No. 123

AM (Applications Manager) 4.0 is a tool for deploying workstation-based client/server applications that access data on LANs and remote systems. It exploits OS/2 for 32-bit, multitasking, multiwindowed applications with interprocess communication and data-sharing. Development can be divided among programmers using reusable components accessed from shared libraries. AM allows development of applications with object-oriented GUIs and ensures well-structured and documented code that can be easily maintained and modified. It provides integration with the DB/2 family, SQL/400, SQL/DS, SQL Server, Supra, and ORACLE. AM also supports MicroDecisionware DB Gateway and TechGnosis SequeLink for access to many other SQL databases. Price: \$8,400 (includes one year telephone support and upgrades).

SQL/Workbench 2.1 is a development tool for designing, compiling, and testing static SQL transactions in an SAA environment. SQL/Workbench produces static SQL packages for IBM databases; it supports DB2, DB2/6000, SQL/400, SQL/DS, DB2/2, and Database Manager. It provides a language-

independent repository that enables users of multiple fourth- and third-generation languages to implement static SQL statements and maintain them separate from applications. Price: \$8,400 (includes one year telephone support and upgrades).

AM Interface for CICS OS/2 1 combines the application manager (AM) development environment with the robust performance and security of CICS. It enables AM-developed applications to access remote data and run transactions, using CICS OS/2 2.0, from a variety of back-ended CICS systems. Price: \$3,000 (Includes one year telephone support and upgrades).

Intelligent Environments, Two Highwood Dr., Tewksbury, Mass. 01876, (800) 669-2797 or (508) 640-1080, fax (508) 640-1090.

INTERSOLV INC. **Circle No. 124**
Excelsior II OS/2, an analysis and design tool for desktop development,

incorporates object-oriented technology and exploits the multitasking and graphical capabilities of OS/2 and IBM's Presentation Manager. A SQL-based LAN Repository offers multi-user support. XL-OS/2 users can implement data-driven, process-driven, and event-driven design methodologies off-the-shelf, or tailor them to local needs with the Customizer. An expert system ensures user input conforms to the implemented methodology. For use with 386, 486, IBM PS/2 or compatible, OS/2 1.3 or 2.0. Price: contact vendor.

Intersolv Inc., 1700 N.W. 167th Pl., Beaverton, Oreg. 97006 (800) 547-4000, fax (503) 645-4576.

M. BRYCE & ASSOCIATES INC. **Circle No. 125**
"Pride" Information Factory 1.1.0, an integrated approach to information resource management (IRM), includes methodologies for enterprise engi-

neering, information systems engineering, and database engineering; an embedded repository to inventory and reuse information resources; and a Project Management system. Its architecture provides the user with the ability to interface with a variety of application development aids (for example, CASE). "Pride" Information Factory is SAA/CUA compliant and translatable. In addition, it uses the Double Byte Character Set (DBCS). Price: begins at \$25,000.

M.Bryce & Associates Inc., 777 Alderman Rd., Palm Harbor, Fla. 34683, (813) 786-4567, fax (813) 786-4765.

MICRORIM INC. **Circle No. 126**
R:BASE 4.5, a 32-bit relational database management system, utilizes the flat memory model to enhance speed and allows multitasking to enable multiple sessions to operate concurrently. Other features include: 4GL with embedded ANSI 89 Level 2 SQL,



MEGA POWER FOR OS/2 BACKGROUND APPS

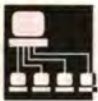
- ▶ Set the priority OS/2 Apps in Full Screen or Window mode including Apps installed as icons. Get information to help prioritize DOS Apps.
- ▶ BIG performance gains in heavy multitasking and/or memory overcommit scenarios or as little as two background Apps via prioritization.
- ▶ Indepth Online Help with Tips, Questions and Answers NEVER BEFORE PUBLISHED Find out how faster machines just "spin" faster and mixing DOS and OS/2 is a disaster.
- ▶ The most powerful Application ever designed for OS/2. It is more effective than multiple processors for Background Apps.
- ▶ We are the steering wheel and gas pedal for OS/2. If you don't have our products—YOU'RE OUT OF CONTROL!
- ▶ "READY FOR OS/2".
- ▶ 32 bit Presentation Manager Application (works on OS/2 2.0 & 2.1). 90-Day money back guarantee! Absolutely required for ALL OS/2 systems. NEW LOWER PRICES.
- ▶ Background Master I/II—32/64 priority levels. \$19/\$29
- ▶ Priority Master I/II—96/128 priority levels. \$39/\$49

ScheduPerformance Inc.

3474 N. University Dr. • Suite 217 • Sunrise, Florida 33351

Voice (305) 486-8299, Fax (305) 486-5018 • M-F 10:00AM to 8:00PM EST





interactive debugger, full text indexing, multicolumn indexing, primary and foreign key support, forms within forms, walkable menus, row-level locking, row and table level entry/exit procedures. Price: \$795.

Microrim Inc., 15395 S.E. 30th Pl., Bellevue, Wash. 98007, (800) 628-6990 or (206) 649-9500, fax (206) 746-9438, CompuServe go microrim.

N SYSTEMS

Circle No. 127

DBAPort 3.0 automates the conversion, duplication, and migration of DDL schema, data, indexes, views, and privileges between multiple RDBMSs including SYBASE/SQL Server for MVS, Oracle, DB2/2, Ingres, SQLBase, Informix, XDB, Ingres, Rdb, HP-ALL-BASE, and ODBC. It includes portable character interface and MS-

Windows GUI interface and allows porting of all or subsets of source databases tables, columns, and data. Also, it maps data to default data types in target database that can be modified. DBAPort also automates several database administration and development tasks. Price: \$3,495 plus \$795 for each database target/source.

N Systems, 2300 103rd Ave., N.E., #A, Bellevue, Wash. 98004, (206) 450-0815, fax (206) 827-8113.

ONTOS INC.

Circle No. 128

ONTOS DB 3.0, a distributed object database management system, provides capability for developing and deploying an expanded range of strategic applications. It is an architecture based on objects, supports workgroup applications, and pro-

vides a range of performance tuning controls. The ONTOS DB architecture includes an object model, an active data dictionary, a programmatic data dictionary interface, and class-based storage management. It is written in object-oriented C++. It is priced at \$5,500.

Ontos Inc., Three Burlington Woods, Burlington, Mass. 01803, (617) 272-7110, fax (617) 272-8101, e-mail stone@ontos.com.

OPENBOOKS

SOFTWARE INC.

Circle No. 129

OpenBooks 1.52 is a client/server database access and query system. It enables users to access, query, analyze, and report on data from multiple database servers. Open Books transforms request for business information into SQL and back again. With a click of the mouse, users see raw data transformed into business information. OpenBooks includes a query manager to meter data from the server, and dynamic forms with automatic drill down. OpenBooks runs on SQL Server, DB2, Oracle, Sybase, and other databases. Price: \$995.

OpenBooks Software Inc., 95 Hayden Ave., Lexington, Mass. 02173, (617) 860-8300, fax (617) 860-8399.

ORACLE

Circle No. 130

Oracle7 for OS/2 is a relational database management system (RDBMS) that runs identically on over 80 distinct hardware and operating system platforms. As a 32-bit application, it utilizes the features of OS/2 2.x. Oracle7 for OS/2 comes in a standalone database; a NetWare edition server which includes Oracle's SQL*Net SPX drivers; and a LAN Server edition which includes Oracle's SQL*N Net-BIOS drivers. Additional wide area network connectivity is provided through Oracle's SQL*Net TCP/IP, SQL*Net APPC for OS/2 servers. The SQL*Net software allows the Oracle7 Server for OS/2 to access distributed data from a variety of other Oracle platforms including DEC VAX, IBM

U.S. POSTAL SERVICE STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION

(Required by 39 U.S.C. 3685)

1A. Title of Publication: OS/2 Developer. 1B. PUBLICATION NO.: Applied for. 2. Date of Filing: 1-Oct-93. 3. Frequency of Issue: Bi-monthly. 3A. No. of Issues Published Annually: 6. 3B. Annual Subscription Price: \$39.95. 4. Complete Mailing Address of Known Office of Publication: Miller Freeman Inc., 600 Harrison Street, San Francisco, CA 94107. 5. Complete Mailing Address of the Headquarters of General Business Offices of the Publisher (Not printers): Miller Freeman Inc., 600 Harrison Street, San Francisco, CA 94107, U.S.A. 6. Full Names and Complete Mailing Address of Publisher, Editor, and Managing Editor Publisher: Cathy Passage, Miller Freeman Inc., 600 Harrison Street, San Francisco, CA 94107. Editor: Dick Conklin, 100 NW 51st Street, Boca Raton, FL 33429. Managing Editor: None. 7. Owner: Miller Freeman, Inc., 600 Harrison Street, San Francisco, CA 94107 U.S.A., a wholly owned subsidiary of United Newspapers plc, 245 Blackfriars Road, London, SE1 9UY ENGLAND. 8. Known Bondholders, Mortgages, and other Security Holders Owning or Holding 1 percent or More of Total Amount of Bonds, Mortgages or Other Securities: None. 9. Does Not Apply. 10. Extent and Nature of Circulation: Average No. Copies Each Issue During Preceding 12 Months: A. Total No. Copies (Net Press Run): 36,500. B. Paid and/or Requested Circulation 1. Sales through dealers and carriers, street vendors and counter sales: 3,949. 2. Mail Subscription (Paid and/or requested): 19,135. C. Total Paid and/or Requested Circulation: 23,084. D. Free Distribution by Mail, Carrier or Other Means, Samples, Complimentary, and Other Free Copies: 875. E. Total Distribution: 23,959. F. Copies Not Distributed: 1. Office use, left over, unaccounted, spoiled after printing: 12,541. 2. Return from News Agents: none. G. TOTAL: 36,500. Actual No. Copies of Single issue Published Nearest to Filing Date: A. Total No. Copies (Net Press Run): 37,000. B. Paid and/or Requested Circulation: 1. Sales through dealers and carriers, street vendors and counter sales: 4,067. 2. Mail Subscription (Paid and/or requested): 17,494. C. Total Paid and/or Requested Circulation: 21,561. D. Free Distribution by Mail, Carrier or Other Means, Samples, Complimentary, and Other Free Copies: 250. E. Total Distribution: 21,811. F. Copies Not Distributed 1. Office use, left over, unaccounted, spoiled after printing: 15,189. 2. Return from News Agents: none. G. TOTAL: 37,000. 11. I certify that the statements made by me above are correct and complete. Cathy Passage, Publisher.

Coming in December from the publishers of

OS/2 Developer/Software Development/LAN Magazine/DBMS/

Database Programming & Design/ Dr. Dobb's Journal/UNIX Review and IBM Corp.!

Your search for OS/2 solutions just got easier.

Now you can access the most comprehensive, detailed listings of vendors and applications serving the OS/2 market — all from one source: the IBM OS/2 AND LAN SERVER APPLICATION DIRECTORY.

This is the complete guide to solutions for even the most specialized applications. It keeps developers up to date on the competition and end-users up to speed on what's available and where to find it.

The 300-page directory gives you fully detailed listings that include:

- Company name
- Contact name
- Address
- Phone number
- Product description
- Details on software features
- Price
- Availability date



Applications are conveniently organized by 17 general industry categories in alphabetical order:

- Accounting/personal finance
- Business-specific
- CAD/CADAM
- Communications

- Database
- Desktop publishing
- Development tools
- Device driver
- Electronic mail
- Games/entertainment
- Graphics
- Information management
- Integrated workgroup
- Networks
- Spreadsheets
- Utilities
- Word processing

ONLY \$9.95

When off-press in December 1993, send me ____ copies of the IBM OS/2 AND LAN SERVER APPLICATION DIRECTORY for only US\$9.95 each, plus US\$3.50 shipping/handling. Add US\$2 for shipping to Canada and overseas.

☐ Payment enclosed
(include sales tax in CA, GA, IL, NY, & TX; GST in Canada)
☐ Charge my: ☐ Visa ☐ MasterCard ☐ American Express
Card # _____ Exp. date _____

Name _____
Title _____
Company _____
Address _____
City/State/Prov. _____
Zip/Postal Code _____ Country _____
Phone _____ FAX _____
Signature (REQUIRED) _____

MAIL TO: Daniel Strickland, OS/2 and LAN Server Application Directory,
Directory Dept., Miller Freeman Inc., 600 Harrison Street, San Francisco, CA
94107 USA FAX TO: (415) 905-2239 OR CALL: (415) 905-2728

Plus you get:

- Index of vendors, cross-referenced by product name
- Special separate section of applications certified by IBM's Ready! for LAN Server certification program

Make your solutions-search easy.

Send for the IBM OS/2 AND LAN SERVER APPLICATION DIRECTORY.

OS/2 is a registered trademark of IBM Corp.

Circle Reader Service Number 63



mainframes, UNIX machines, and NetWare file servers. Oracle7 for OS/2 includes new data loading facilities as well as National Language support. Also, available separately are Oracle's precompilers (Pro*C, Pro*COBOL, and Pro*Fortran) which can be used to develop OS/2, Windows, or DOS client application. Price: Oracle7 Database for OS/2 is \$699; call for pricing information for Oracle7 Server.

Oracle Corporation, 400 Oracle Pkwy., Redwood Shore, Calif. 94065, (415) 506-5888, fax (415) 506-7206.

PROTOVIEW DEVELOPMENT CORPORATION

Circle No. 131

SQLView 1.0 Visual Database Access provides live data manipulation of over 30 different database sources through the ProtoGen+ Workbench—including a database that writes to ODBC or Q+E Library (Q+E data drivers included). This product requires ProtoGen+. Price: \$395.

ProtoView Development Corporation, 353 Georges Rd., Dayton, N.J. 08810, (800) 231-8588 or (908) 329-8588, fax (908) 329-8624, CompuServe 75300,3510.

Q+E SOFTWARE

Circle No. 132

Q+E Database Editor 5.0, an intuitive ad hoc query and reporting tool, generates reports with breaks and subtotals, accesses databases identically, builds queries, designs query by example forms, links data into other applications, and has a customizable Icon Bar. Requirements: IMB PS/2, PC/AT or compatible, 2MB RAM, 4MB Hard Disk space. Price: \$299.

Q+E Database Library 2.0 for OS/2 provides a tool for building ODBC-enabled applications. It provides SQL and PC DBMS client access using ODBC for OS/2. It also provides functions for transaction processing, data dictionary, stored procedures, parameter for creating multipurpose SQL queries, SQL statements and parameterized queries, data conversion, and column binding. Q+E Database Library 2.0 for OS/2 includes the

Q+E Query Builder to provide users with a way to build SQL statements and ODBC-compliant Q+E Drivers. Q+E Database Library 2.0 for OS/2 and the supplied Q+E Drivers are 32 bit. Price: \$699. The Q+E Database Library 2.0 will ship on January 31, 1994.

Q+E Software, 5540 Centerview Dr., Suite 324, Raleigh, N.C. 27606, (800) 876-3101 or (919) 859-2220, fax (919) 859-9334, CompuServe 71333,125.

SAS INSTITUTE INC.

Circle No. 133

The SAS System 6.0 provides tools to access, manage, analyze, and present data within an applications development environment. Capabilities include EIS, spreadsheets, graphics, data analysis, report writing, quality improvement, project management, computer performance evaluation, client/server computing, database access, decision support, and applications development. The software provides an object-oriented development environment and offers a code-free development environment that consolidates the SAS System's capabilities for data access, management, analysis, and presentation. The software includes procedures for controlling the users path through an application and transporting screens between operating environments. Price: licensing fee schedule for Base SAS on one PC is \$940 for the first year, \$430 for renewal. Call for full license fee schedule.

SAS Institute Inc., SAS Campus Dr., Cary, N.C. 27513, (919) 677-8000, fax (919) 677-8123.

S-CUBED INC.

Circle No. 134

AnyClient 1.0 Programmable Middleware enables the user to update multiple databases from queries displayed on spreadsheets, project managers, word processors, visual development environments, or tools users have chosen to display their queries. AnyClient permits clients to query and update multiple tables simultaneously. It traps illegal multiple updates and

violations of database rules, transforms them into valid updates, and processes the data. Users are insulated from the complexities; a DBA sets up the rules that create a virtual database and AnyClient does the rest, permitting download and upload of data to or from a client environment (Excel, Lotus, Access, and so on). Clients that previously could launch only SQL commands now can take advantage of RPC features and functions of the underlying communication medium and RDBMS. Call for pricing information.

DB Manager C/S Plus 1.0 provides developers of DB2/2 and DB6000 based systems with an ODBC compliant driver that takes advantage of DARI/DRDA and optionally provides them with a transaction oriented fourth generation language. DB Manager C/S Plus enhances the functionality of DB2/2 and DB6000 by providing them with stored procedure and trigger policy functions. It also provides client side ODBC compliance with stored procedure extensions. Price: \$795-developer, \$195-server engine. DB Manager C/S Plus will be ship on March 15, 1994.

S-Cubed Inc., 1010 Washington Blvd., Stamford, Conn. 06901, (203) 323-0760, fax (203) 323-2007.

SEQUIER

SOFTWARE INC.

Circle No. 135

CodeBase 5.0 and **CodeBase++ 5.0** are DBMS source code libraries that permit database management with multiuser dBASE, FoxPro or Clipper file compatibility from C, C++, Pascal, or Basic. They have bit optimization query technology that analyzes queries using index information, so a record is only retrieved from disk when it belongs in a query solution set. The user can utilize CodeReporter to design reports interactively and then generate corresponding source code. CodeBase and CodeBase++ allow porting between DOS, Windows, OS/2, Macintosh, and UNIX. Price: \$495, \$1090 multiplatform version for UNIX and Macintosh.

CodeTranslator 2.1 automatically translates dBASE III PLUS and Clipper Summer '87 applications into C. It allows the user to quicken a slow application by converting it to C and using it with CodeBase. Applications look and act like the original programs. Use CodeTranslator to convert existing applications or to help learn C and CodeBase. Price: \$245.

Sequiter Software Inc., P.O. Box 575, Newmarket, N.H., 03857-0575, (403) 437-2410, fax (403) 436-2999.

SYBASE INC. Circle No. 136

Sybase SQL Server 10.0 is a relational database server based on multi-threaded database operating system architecture. It features scalable, server-enforced integrity, application availability, and on-line distributed RDBMS capabilities. It provides the user with connectivity to link any terminal, personal computer, or workstation with a server. SQL Server technology controls and supports open, standards-based connectivity. Price: \$3,750-158,040.

OmniSQL Gateway gives users transparent access to corporate data regardless of where or in what kind of database the information resides. Non-SYBASE data can now look and feel like SYBASE data, and the user can work with it using the industry-leading functionality of the SYBASE SQL Server. OmniSQL Gateway can be used for customized applications, data migration, and ad hoc decision support. OmniSQL Gateway consists of two components: (1) an OmniSQL server available for OS/2, major UNIX platforms, VAX VMS, and Novell NetWare; and (2) OmniSQL Access Modules that exist on the platform of the target source. OmniSQL Access modules currently exist for ORACLE, SYBASE, DB2, RMS, ISAM, and other enterprise-wide databases. Price: \$2,250-83,000 for OmniSQL Server. \$2,250-102,730 for OmniSQL Access Modules.

SYBASE Open Server 2.0 development toolkit allows developers to build server applications that appear

to Sybase Open Client users as though they were a multithreaded SYBASE SQL Server. Organizations can extend the data access capabilities of their Sybase Open Client and DB-Library-based client applications beyond Sybase SQL Server data to the data made available through Sybase Open Server applications regardless of the data type, source, or location. A data source that is accessible to the application programmer (robotics, telephone switching systems, process control sensors, stock quote feeds, room monitoring devices, and so on) can be accessed using Open Server. Open Server also shields server applications from the communications protocols of the network on which they are operating. Using the Open Server toolkit, programmers create applications that run on many different transports. SYBASE Open Server for CICS provides users on a network with access to IBM mainframe data and services. This includes MVS data, applications, and mainframe services running under the control of IBM's CICS transaction processing monitor. It extends the huge investments in mainframe applications that access VASM, DL/I, DB2, and third party DBMSs by making them available as servers for SYBASE client applications. Price: \$445-24,950 for PC DOS or OS/2.

SYBASE Open Client 4.6 is an application toolkit that provides programmers with the services to develop applications that access diverse data sources. Its run-time services free developers from dealing with a variety of application protocols and networking issues. It supports multiple application programming interfaces (APIs) such as Sybase Embedded SQL, SYBASE DB-Library and the Microsoft ODBC (Open Database Connectivity). This allows applications that utilize Open Client to read/write access to Sybase SQL Server data as well as non-Sybase data via the Sybase OmniSQL Gateway or through Sybase Open Server. Open Client also supports a number of transport protocols via Open

Client's Net Library. Price: \$470-17,280 list price.

Sybase Inc., 6475 Christie Ave., Emeryville, Calif. 94608, (800) 8SYBASE or (510) 596-3500, fax (510) 658-9441.

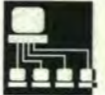
VERSANT OBJECT TECHNOLOGY CORPORATION Circle No. 137

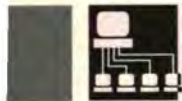
VERSANT ODBMS 2.3 is a ODBMS designed to support multiuser, production applications in the distributed environment. VERSANT is built within a scalable distributed architecture that provides features such as transparent data distribution, object-level locking, dynamic schema evolution, and built-in workgroup computing support. Price: \$6,000 and up.

Versant Object Technology Corp., 1380 Willow Rd., Menlo Park, Calif. 94025, (800) VERSANT (837-7268) or (415) 329-7500, fax (415) 325-2380.

XDB SYSTEMS INC. Circle No. 138

XDB QMT 3.0 is an IBM query management facility (QMF) compatible, PC and workstation-based data query and reporting facility designed to operate with mainframe DB2 or local database engines. XDB-QMT can be used as an off-line tool for report development, testing, and training, or as an on-line tool with XDB-Link for real-time production queries and reports. Using XDB-QMT strictly as an off-line tool, the user can download a representative sample of a mainframe database to the DB2 compatible XDB-Server on a LAN or to an individual PC. Report objects, such as queries, report forms, and procedures, created with XCB-QMT can be executed on the PC or exported to QMF on the mainframe, without modification, for production against DB2 data. XDB-QMT can be used as a real-time, on line tool when it is configured with XDB-Link. When using XCB-Link, XDB-QMT users can access mainframe DB2 data directly from a PC in a stand-alone environment or LAN configuration. Price: \$1,200. Version 3.3 will be shipping First Quarter, 1994.





XDB's SQL 3.0 engine provides referential integrity, disturrancy control, and system catalogue. XDB-SQL C SDK comes complete with a C precompiler and a C-API. C programmers can embed SQL into a program or call SQL engine functions using a C function call library. Application written using XDB-SQL C SDK can be used either in a stand-alone environment against a resident SQL engine, as a client workstation accessing any XDB-Server on a LAN, or connected to mainframe DB2 via XDB-Link. Price: \$650. Version 3.3 will be shipping First Quarter, 1994.

XDB-Server 3.2 is a 32-bit, multiuser database system that features referential integrity, distributed database access, transaction processing, backup and recovery, authorization, concurrency control, and system catalogue tables. New features with the 3.3 version (available First Quarter, 1994) include compatibility with DB2 3.0, query optimization enhancements in read-only isolation levels, a governor that allows the user to limit the number of record retrieved, and server-to-server communications. Clients running under Windows NT, Windows, OS/2, and DOS provide user interfaces that gather input and prepare requests for the XDB-Server. The XDB-Server provides

storage, access, and control of data. Because the XDB-Server is network-independent, DB2 compatibility can exist on multiple platforms including Windows NT and OS/2. Any of XDB's development, fourth generation language, and reporting tools, as well as third party tools including PowerBuilder, can act as clients to the XDB-Server. Price: \$2,495 (1-5 users; \$100 each additional users). Version 3.3 will be shipping First Quarter 1994.

XDB-Link 3.2 contains a set of interface software that links SQL applications on PCs, LANs, and workstations to DB2 data residing on the mainframe. XDB-Link consists of two components: XDB-Link DRDA Gateway, which resides on the PC, and XDB-Link host-based CICS program, which resides on the mainframe. These components communicate via LU 6.2 protocols to satisfy DB2 SQL requests made from PCs, LANs, and workstations. XDB-Link is based on a client/server architecture. It supports DOS, OS/2, and Windows clients. In a development environment, XDB-Link extends systems level testing to the PC environment by providing access to mainframe production data. In a production environment, XDBs application tools and LAN applications can now access DB2

data residing on the mainframe. XDB-Link provides access to IBM's relational databases, including DB2, DB2/VM, and SQL/400 via DRDA. In addition, XDB-Link provides access to non-relational databases via Remote Procedure Calls (RPCs). Price: \$8,500 (1-10 users). Version 3.3 will be shipping First Quarter 1994.

XDB-Workbench 3.0 is a DB2-compatible development and environment for PCs, LANs, and workstations. The product consists of a powerful SQL database engine, a COBOL precompiler, and a suite of development and administrator utilities. The COBOL precompiler follows SAA standards. The XDB-Workbench enables the user to develop, test, and run COBOL programs with embedded SQL on PCs, LANs, and workstations. The user migrates the programs to the mainframe. No source modification is required. The XDB-Workbench can be used stand-alone, as a client accessing any XDB-Server on a LAN, or connected to mainframe DB2 via XDB-Link. Price: \$1,495. Version 3.3 will be shipping First Quarter, 1994.

XDB Systems Inc., 14700 Sweitzer Ln., Laurel, Md. 20707, (800) 488-4948 or (301) 317-6800, fax (301) 317-7701, email xdb@xdb.com.



New Tools and Utilities



TOOLS

AM/Harvest. AM/Harvest delivers capabilities for building client/server applications and managing the sharing, reuse, and maintenance of application code. AM/Harvest simplifies project management: code is shared, reused, and maintained within the development teams. It addresses the complexities of managing change during iterative client/server development processes. AM/Harvest runs on OS/2 2.0 or later and works with Intelligent Environment's AM 3 or higher. It costs \$5,500.00 per server, including one year of free technical support and upgrades.

Intelligent Environments
Phone: (508) 640-1080, Fax: (508) 640-1090

Device Driver Source Kit for OS/2. A CD-ROM of OS/2 device driver source code, IBM's Device Driver Source Kit enables hardware vendors to create their own OS/2 device drivers. The kit contains source code for over 79 device drivers and tools, along with on-line reference books that include information on OS/2 device drivers and device driver function, and hints and techniques for device driver construction. It supports various device types, including display, printer, CD-ROM, direct access storage device, small computer system interface, disk, mouse, keyboard, multimedia, pen, touch, serial, and parallel. Device Driver Source Kit is priced at \$499.00.

IBM Corp.
Phone: (800) 633-8266

DocuMento 1.1. A 32-bit development tool that works with OS/2 2.0 and 2.1, DocuMento allows developers to create on-line documentation and help screens without GML skills. Using DocuMento, developers work in a parallel fashion, docu-

menting essential information while writing code. Features include paragraph styles, two-level indexes, hypertext links, fonts, color, embedded graphics, text emphasis, interface to PM programming and Gpf, assistance through Lotus Ami Pro macros, and various output formats.

SE International Inc.
Phone: (800) 473-4685 or (407) 241-3428

Galaxy Application Environment 2.0. Galaxy Application Environment 2.0 is a cross-platform development environment for creating applications that are graphical and distributed. Galaxy Release 2.0, available in C and C++ versions, adds new features, and establishes a consistent set of application development capabilities across supported platforms.

Visix Software Inc.
Phone: (800) 832-8668 or (703) 758-8230,
Fax: (703) 758-0233

GammaTech Utilities for OS/2. Release 2.1 of GammaTech Utilities' OS/2 workstation software provides the ability to perform volume recovery, optimization, and maintenance operations without extensive technical knowledge. The utilities support high performance file system and file allocation table formats. Many of the features operate in both the Presentation Manager and Command Line modes.

SofTouch Systems Inc.
Phone: (405) 947-8080

IPF Editor for OS/2 2.x. A 32-bit PM IPF development editor, IPF Editor provides tools for creating on-line help and on-line documents for OS/2 2.x. Designed to be easy to use for both programmers and non-programmers, it provides a method for



adding help to applications or converting existing documents into on-line documents. To take advantage of all IPF Editor functions, it must be used with IBM OS/2 Toolkit 2.x's IPF Compiler. IPF Editor is available for \$95.00.

Perez Computing Services
Phone: (206) 428-5025

Pen for OS/2. A pen computing system that adds pen capabilities to OS/2, DOS or DOS/Windows applications, Pen for OS/2 is designed for mobile computing desktops and for collaborative computing on the desktop. Capabilities include handwriting recognition, a window that adds handwriting recognition to non-pen-aware applications, standard and user-customizable gestures, and a pop-up keyboard. Included in the package are Telepen, a collaborative computing system, and Sketchpad, a freehand drawing tool. Pen for OS/2 requires a computer running OS/2 2.1, a pen sensor (digitizer) integrated with a display or digitizing tablet, 2MB RAM over the requirements of OS/2 2.1, and 5MB of hard disk space. The single unit price is \$89.00.

IBM Corp.
Phone: (800) 342-6672

System Architect 3.0. A new version of System Architect, a CASE tool for developers building client/server applications, offers network security, access control, enhanced data modeling, IDEF0 and IDEF1X support, a graphical user interface, object-oriented features, color, real-time modeling, and a browser. The OS/2 PM 2.1 version costs \$1,795.00 for the single user (merge) version and \$3,790 for the first two copies of the network version.

Popkin Software & Systems Inc.
Phone: (800) 732-5227, Fax: (212) 571-3436

Team/V 1.2. An extension to the Smalltalk/V object-oriented environment, Team/V allows development teams to organize, structure, and coordinate their work. Because the releases for Win32 and OS/2 have the same interface, programmers using Team/V on OS/2 and Windows 3.1 or Windows NT can share common repositories and develop packages that

may be used in either environment. Team/V provides tools to define, store, browse, and share packages, which are modules that define a program function or element. Packages contain source code for Smalltalk/V definitions; they also can contain descriptive information such as comments, author name, and version number. Team/V is priced at \$1,595.00. Registered Smalltalk/V users can buy Team/V at a special price for a limited time.

Digitalk Inc.
Phone: (714) 513-3000, Fax: (714) 513-3100



UTILITIES

Advanced Task Scheduler for OS/2. A job scheduler utility, Advanced Task Scheduler (ATS) is designed to allow developers to build job streams, giving them complete control of how and when to run each program. ATS lets developers set the time, day, and month a job should run, among other features. It allows developers to define an unlimited number of tasks, dependencies, and holidays, and logs all activity on disk. ATS for OS/2 requires OS/2 2.0 or later. It costs \$249.00; quantity discounts and site licenses are available.

MHR Software and Consulting
Phone: (908) 821-0359

OS/2 CALENDAR

March 15-17	Software Development, San Jose, Calif. (800) 441-8826
April 19-21	OS/2 Technical Interchange, San Francisco (407) 982-9353
May 2-6	PEN EXPO, Boston (508) 649-4200
July 19-22	OS/2 World, Santa Clara, Calif. (415) 905-2354

CodeBase 5.0

New for OS/2

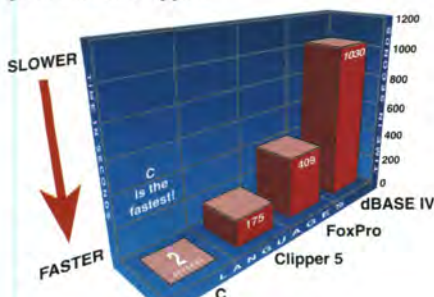
Discover why FoxPro, Clipper, and dBASE were all written in C.

There is a good reason why your database language was developed in C. In fact, there are many good reasons.

C code is small. C code is fast. C code is portable. C code is flexible. C is the language of choice for today's professional developer. With the growing complexity of database applications, C is a realistic alternative. Now with CodeBase 5.0, you can have all the functionality, simplicity and power of traditional database languages together with the benefits of C/C++.

C speed - fast code, true executables...

FoxPro, Clipper, and dBASE were written in C primarily for speed. But those compilers don't really compile, they combine imbedded language interpreters into your .EXE. Now that's slow. For dazzling performance you need the true executables of C. With CodeBase you get the real thing, C code. Consider the following statistics, from the publisher of Clipper:



"Sieve of Erasthothenes"
Benchmark for Prime Number Generation
Shows C to be incredibly faster!

C size - small executables, no added overhead...

FoxPro, Clipper and dBASE would like you to believe you need their entire development system to build database applications. But

remember, those products are all written in C. So why do you need to lug all their extra code around? You don't. CodeBase is a complete DBMS, in C. No fat executables stuffed with unused code. No runtime modules. No royalties. Just quality C code. CodeBase is just what you need.

C portability - ANSI C/C++ on every hardware platform...

No other language exists on more platforms than C/C++. Why rewrite your entire application for DOS, Windows, Windows NT, OS/2 or UNIX? With CodeBase the complete C source code is included, so you can port to any platform with an ANSI C or C++ compiler. Now and in the future.

dBASE Compatible data, index and memo files...

You want the industry standard. You need compatibility. Sure, dBASE is the standard, but every dBASE compatible DBMS product uses its own unique index and memo file formats. Only CodeBase has them all: FoxPro (.cdx), Clipper (.ntx), dBASE IV (.mdx) and dBASE III (.ndx). Now it's your choice, we're compatible with you.

Announcing CodeBase 5.0

The power of a complete DBMS, the benefits of C

NEW - Multi-user sharing with FoxPro, Clipper and dBASE...

Now your multi-user C/C++ programs can share data, index and memo files at the same time as concurrently running FoxPro, Clipper and dBASE programs. No incompatibilities. No waiting.

NEW - Queries & Relations 1000 times faster...

CodeBase 5.0 now lets you query related

data files with any logical dBASE expression. Our new Bit Optimization Technology (similar to FoxPro's Rushmore technology) uses index files to return a query on a 1/2 million record data file in just a second. Automatically take advantage of this query performance by using our new CodeReporter:

Product Sales Summary		
Month at:	Nov. 1992	
Product	Quantity	Value
Cassette	63	\$25,137.00
Spread Sheet	58	\$21,986.00
Monthly Summary	121	\$47,063.00

Product Sales Summary		
Month at:	Dec. 1992	
Product	Quantity	Value
Cassette	62	\$24,862.00
Spread Sheet	53	\$19,875.00
Monthly Summary	115	\$44,737.00

Summary		
Product	Quantity	Value
Cassette	62	\$24,862.00
Spread Sheet	53	\$19,875.00
Monthly Summary	115	\$44,737.00

To use CodeReporter, simply draw your report, then include it in any program you write. Call 403/437-2410 now for your FREE working model of CodeReporter.

New - Design complex reports in just minutes...

Our new CodeReporter takes the painstaking work out of reports. Now simply design and draw reports interactively under Windows 3.1, then print or display them from any DOS, Windows or UNIX application.

SPECIAL - FREE CodeReporter

Order CodeBase 5 before June 30, 1993 and receive CodeReporter for free! This offer includes our no-risk, 90-day money back guarantee, so order today!



CodeBase 5.0
The C/C++ Library for DataBase Management

Call Now
403-437-2410

SEQUITER SOFTWARE INC. FAX 403-436-2999
Europe 33.20.24.20.14
#209.9644-54 AVE., EDMONTON, AB, CANADA T6E-5V1

©1992 Sequiter Software Inc. All rights reserved. CodeBase is a trademark of Sequiter Software Inc. All other trade names referenced herein are property of their respective companies. MAdvertising by MicroArts



New Borland C++ 4.0 Visual is just the beginning

Why settle for ordinary visual when new Borland C++ gives you so much more? The highest quality fourth-generation tool set. A fully customizable and open desktop. The ability to target

16- and 32-bit Windows simultaneously. And, of course, it's "visual."

Only Borland C++ gives you a fully integrated professional editor featuring BRIEF® technology, powerful Turbo Debugger® GX, and integrated C and C++ VBX control support.

An environment to die for

Borland C++ 4.0 takes the bureaucracy out of development. With the visual Project Manager and its multi-target capabilities, even the most complex projects are handled automatically.

The flexibility of AppExpert actually

generates much of your application for you. TargetExpert, ClassExpert, DialogExpert, and Resource Workshop™ streamline the tasks of setting up and customizing your applications.

A true C++ implementation that's years ahead

Languages evolve for a good reason—so programmers can realize ever-increasing productivity and safety of code. That's why Borland is the first to bring important new enhancements to the C++ language, like full support for templates, exception handling, and Run-time Type Information. And Borland C++ includes ObjectWindows™ Library (OWL) 2.0—the world's most popular framework.

Borland C++ is the world-standard C++ because it's the easiest to use, yet has power to spare for the most complex tasks. If you're serious about C++ programming, new Borland C++ 4.0 is the environment you've got to

have. After all, if your compiler is only visual, it's just a facade. Get new Borland C++ 4.0 for Windows and DOS now.

Borland C++ owners
Upgrade Now!

\$149⁹⁵ (Suggested list price \$499.95)

Other C++ owners

\$199^{95*}

90-day, money-back guarantee!

**See your dealer or call now,
1-800-336-6464, ext.7101**

In Canada call, 1-800-461-3327.

Borland

Power made easy™

G362-0001-21

Copyright © 1993 Borland International, Inc. All rights reserved. All Borland product names are trademarks of Borland International, Inc. *Offer good for owners of Microsoft and Symantec C or C++ products; also Turbo C++ owners who purchased product before November 1, 1993. Offer good in the United States and Canada only. All prices in U.S. dollars. Dealers prices may vary. BI 5827

Circle Reader Service Number 65

